PacketCable™ Distributed Call Signaling Specification

PKT-SP-DCS-D03-000428

Draft Work in Progress

Notice

Document Status Sheet

Document Control Number:	PKT-SP-DCS-D03-000428			
Document Title:	PacketCable™ Distributed Call Signaling Specification			
Revision History:	D01-990812: First release for Vendor Comments D02-991007: Release for IETF SIP-WG Comments D03-000428: Release for Vendor Comments			
Reference:	PacketCable™ Distributed Call Signaling Specification			
Responsible Author:	Packetcable-Dcs Focus Team, Bill Marshall, Editor			
Status:	~~Work in Progress~~	Draft	~~Interim~~	~~Released~~
Distribution Restrictions:	~~Author Only~~	~~CL/Member~~	~~CL/ PacketCable/ Vendor~~	Public

Key to Document Status Codes:

Work in Progress	An incomplete document, designed to guide discussion and generate feedback, that may include several alternative requirements for consideration.
Draft	A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
Interim	A document which has undergone rigorous Member and vendor review, suitable for use by vendors to design in conformance to and for field testing.
Released	A stable document, reviewed, tested and validated, suitable to enable cross-vendor interoperability.

Table Of Contents

List Of Figures

1. Introduction

1.1 Scope

This specification describes the PacketCable Distributed Call Signaling (DCS) protocol intended for use by PacketCable Call Management Servers (CMS) to communicate with other call management servers to support packet-based voice and other real-time multimedia applications. The CMSs may support endpoints that use the protocol specified by the Network based Call Signaling (NCS) protocol for communication between the endpoint and the CMS. This specification also covers the protocol used by PacketCable endpoints to support packet-based voice, video, and other real time multimedia services. PacketCable endpoints that may use this protocol include:

- ◊ Media Terminal Adapters (MTA) supporting traditional analog phones. MTAs may be embedded in a DOCSIS 1.1 cable modem or attached to a local area network supported by cable modem.
- ◊ Other (intelligent) end-points such a personal computers.

This document specifies a profile of the Session Initiation Protocol 2.0 (SIP) that includes a set of SIP extensions and usage rules that support commonly available local and CLASS services. DCS may also take advantage of endpoint intelligence in supporting telephony services and features, while at the same time providing flexibility to allow evolution in the services that may be provided by these endpoints.

DCS takes into account the need to manage access to network resources and account for resource usage. The usage rules defined in this specification specifically address the coordination between Distributed Call Signaling and PacketCable Dynamic QoS mechanisms for managing resources over the cable access network. In addition, the DCS specification defines the protocols and messages needed between call management servers for supporting these services.

This document does not specify the protocols or procedures between service providers that do not maintain mutual trust relationships. This document does not specify the interaction between a user (human) of the packet-based service and the MTA or other end-point. Various terms such as offhook, onhook, etc., are informational only, and serve to illustrate the DCS mechanisms by comparison to a commonly used telephone instrument.

Other PacketCable documents describe interfaces between other system elements, such as Event Message Recording [3], Dynamic Quality of Service [4], Operations and Provisioning [5], Electronic Surveillance [10], and Security [2]. These other specifications place requirements on the signaling protocol to transport various pieces of information needed to implement a complete system. The current document specifies the syntax and protocol procedures that implement these requirements.

1.2 Specification Language

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"MUST"	This word or the adjective "REQUIRED" means that the item is an absolute requirement of this specification.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this specification.

“SHOULD”	This word or the adjective “RECOMMENDED” means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
“SHOULD NOT”	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
“MAY”	This word or the adjective “OPTIONAL” means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

2. Background and Motivation

The design of the Distributed Call Signaling (DCS) architecture recognizes the trend towards use of packet networks as the underlying framework for communications. These networks will provide a broad range of services, including traditional best-effort data service, as well as enhanced value-added services such as telephony. The Network based Call Signaling (NCS) protocol may be used to communicate between limited function end-points, such as standard telephone sets, and Call Management Servers (CMS). The NCS protocol specification covers the communication between such end-points and the CMS that controls them, and does not address the issues that may arise for communication between multiple CMSs that may reside even within a single service provider's network. Therefore, this specification covers the signaling performed between CMSs. The initial real-time multimedia service that is supported by the Network based Call Signaling function is that of interactive telephony.

We recognize that packet based networks may also offer additional real-time multimedia services to endpoints that are IP capable. Also, improvements in silicon will reinforce the trend towards increased functionality in endpoints. These intelligent endpoints will take advantage of the widespread availability of packet networks to enable a rich set of applications and services for users.

When the network is used for real-time telephony applications, it is essential to have service differentiation at the IP layer. The ability to control and monitor usage is needed for the provider to be able to provide service differentiation and to derive revenue from enhanced services. At the same time, the availability of best effort communications and the migration of functionality to the endpoints poses a challenge to the provider to find incentives for users to use or pay for enhanced services.

There are three key functions that a provider can offer as incentives to use enhanced services. First, the network service provider has the unique ability to manage and provide network layer quality of service. When users depend on the quality of the service, as with telephony, there is a strong incentive to use the enhanced service rather than a best effort service. Second, the network service provider can play an important role as a trusted intermediary. This includes ensuring the integrity of call routing, as well as ensuring both the accuracy and the privacy of information that is exchanged. The service provider can also add value by ensuring that services are provided consistently and reliably, even when an endpoint is unavailable. Finally, there are a number of services that may be offered more efficiently by the network service provider rather than in endpoints. For example, conference bridging may be more cost effective to implement in a multi-point bridge rather than in every endpoint attached to the network.

A key contribution of the DCS architecture is a recognition of the need for coordination between call signaling, which controls access to telephony specific services, and resource management, which controls access to network-layer resources. This coordination is designed to meet the user expectations and human factors associated with telephony. For example, the called party should not be alerted until the resources necessary to complete the call are available. If resources were not available when the called party picked up, the user would experience a call defect. In addition, users expect to be charged for service only after the called party answers the phone. As a result, usage accounting starts only after the called party picks up. Coordination between call signaling and resource management is also needed to prevent fraud and theft of service. The coordination between DCS and Dynamic QoS[4] protocols ensures that users are authenticated and authorized before receiving access to the enhanced QoS associated with the telephony service.

When the endpoints are limited in functionality, both in processing and storage, the network based call signaling protocol (NCS) [8] is suitable. The protocol functionality required of the endpoint is simple, and more of the functionality resides in the network in call management servers. In the context of interactive telephony, the state of a call is maintained within the CMS. The CMS is responsible for controlling the establishment and manipulation of call legs and for requesting and obtaining network layer QoS for the call. The NCS protocol specifies the information and message exchange between the endpoint and the

CMS. When the call has to be routed through multiple CMSs, additional functionality is required in the protocol, to communicate the information related to the call. This information includes that provided by the endpoint to the network as well as information that may reside in the CMS or other entities within the network that relates to the call. Examples of such additional information that may reside in the network include billing and information that may otherwise be kept private from untrusted endpoints.

It is important to be able to deploy a residential telephone service cost-effectively at very large scale. When the endpoints are intelligent and have adequate processing and storage capability, we would like to exploit their capability to enable the service to scale up. To achieve this, DCS minimizes the messaging overhead on network call servers, and does not require these servers to maintain state for active calls. Once a call is established, call state is maintained only where it is needed: at the intelligent endpoints that are involved in the call, and at the CMTSs in the bearer path that are providing differentiated service to the media flow. This allows the network call servers to scale to support more users, and imposes less stringent reliability requirements on those servers.

DCS is also designed so that calling users receive consistent service even when a called endpoint is unavailable. For example, when an endpoint is unavailable, service logic in a network call server can forward telephone calls to a voice mailbox.

2.1 Requirements and Design Principles

In this section, we briefly describe the application requirements that led to a set of DCS signaling design principles. In its most basic implementation, DCS supports a residential telephone service comparable to the local telephone services offered today. In addition to the service features that need to be supported, as described in Section 2.3, there are important requirements in the areas of reliability, performance, and scalability that influence the signaling architecture. First line telephony service requires enhanced bearer channel and signaling performance, including:

> Low delay – end-to-end packet delay must be small enough that it does not interfere with normal voice conversations. The ITU recommends no greater than 300 ms roundtrip delay for a telephony service.
>
> Low packet loss – packet loss must be small enough to not perceptibly impede voice quality or performance of fax and voice band modems.
>
> Short post-dial delay – the delay between the user dialing the last digit and receiving positive confirmation from the network must be short enough that users do not perceive a difference with post-dial delay in the circuit switched network or believe that the network has failed.
>
> Short post pickup delay – the delay between a user picking up a ringing phone and the voice path being cut through must be short enough so that the “hello” is not clipped.

We identify a number of key design principles that arise from the requirements and philosophy outlined above.

1. Providing differentiated network-layer quality of service is essential, while allowing the provider to derive revenues from the use of such differentiated services.

2. The DCS signaling architecture should allow for communication between CMSs in the network. At a high level, one may consider a CMS to perform complex signaling tasks on behalf of an endpoint. When the network includes multiple CMSs, DCS should provide the call signaling function between the CMSs on an individual call basis. Within such a context, the DCS architecture should allow the network to support limited function endpoints, while allowing the additional functions to be performed by CMSs, including the maintenance of call state in the CMS.

3. The architecture must ensure that the network is protected from fraud and theft of service. The service provider must be able to authenticate users requesting service and ensure that only those authorized to receive a particular service be able to obtain it.

4. The architecture must enable the service provider to add value by supporting the functions of a trusted intermediary. This includes protecting the privacy of calling and called party information, and ensuring the accuracy of the information that is provided in messages from the network.

5. The architecture must enable the service provider to give a consistent view of basic services and features even when customer premise equipment is unavailable, and allow users to take advantage of functionality that is provided in the network, when it is cost-effective and easy to use.

6. The architecture should allow, and even encourage, implementation of services and features in the intelligent endpoints, where economically feasible, while still retaining value in the network and network-based services

7. The architecture must be implementable, cost-effectively, at very large scale.

2.2 Distributed Call Signaling Architecture

The Distributed Call Signaling Architecture follows the principles outlined above to support a robust telephony service. Figure 1 introduces the key components in the architecture.

The architecture assumes a broad range of DCS-compliant endpoints that provide telephone service to the user including *Media Terminal Adapters* (MTAs) that may be integrated with a Cable Modem or standalone, as well as other endpoints such as personal computers. The cable access network interfaces to an IP backbone through a CMTS. The CMTS is the first trusted element within the provider's network. It performs resource management and acts as a policy enforcement point and source of billing information.

When the MTA uses the Network based Call Signaling protocol, the Call Management System (CMS) performs several functions. It uses the NCS protocol to interface with the MTA. A CMS that establishes and receives calls on behalf of an endpoint is referred to as a CMS/Agent in this specification. The CMS/Agent uses the protocol specified here to communicate with other CMSs. In addition, it may also perform the function of a Gate Controller (GC), which is responsible for authorizing the enhanced quality of service for the media stream.

In the DCS framework supporting intelligent endpoints, the *Call Management System* (CMS) consists of two components, a *CMS/Proxy* and a *Gate Controller* (GC). A CMS/Proxy processes call signaling messages and supports number translation, call routing, feature support and admission control. In the context of SIP [11], a CMS/Proxy is a SIP proxy that is involved in processing and forwarding of SIP requests. A CMS/Proxy act as a trusted decision point for controlling when resources are committed to particular users. *Gate* Controllers (GC) manage access to enhanced QoS. *Media servers* represent network-based components that operate on media flows to support the service. Media servers perform audio bridging, play terminating announcements, provide interactive voice response services, etc. Finally, *PSTN gateways* interface to the Public Switched Telephone Network. The CMS/Proxy and the Gate Controller may reside on separate physical entities, or may be implemented in a single physical entity. In this specification, we do not expose the interface between the CMS/Proxy and the Gate Controller.

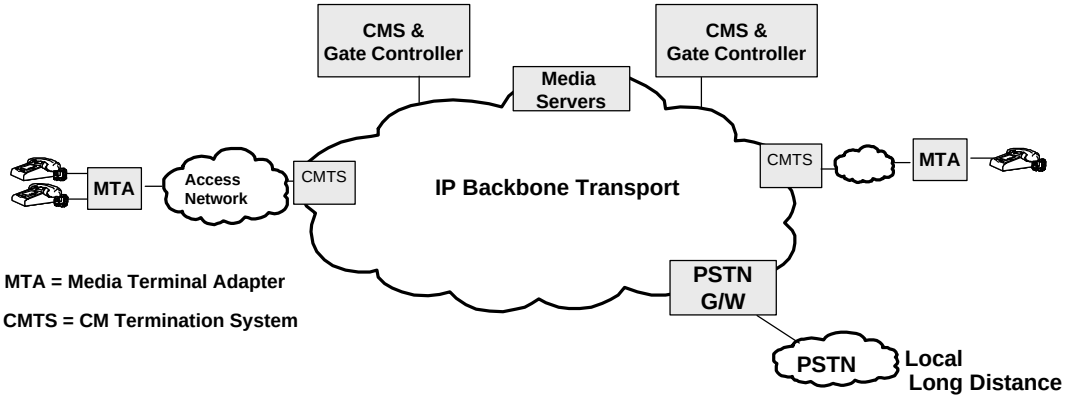


Figure 1: System Architecture

Telephony endpoints are considered to be "clients" of the telephony service. When the endpoints are limited in function (such as standard telephone sets), the network-based call agents may be responsible for providing services. However, with intelligent endpoints, the architecture allows a range of services to be implemented by these intelligent endpoints. Intelligent end-points may collect dialed digits, participate in signaling and contain the service logic required for basic call setup and feature support. Endpoints also participate in end-to-end capability negotiation. However, endpoints are not trusted to provide accurate information to the network or to keep information private, except when it is in the endpoint's best interests to do so.

Access to network resources must be controlled by the service provider. The CMTS receives resource management requests from endpoints, and is responsible for ensuring that packets are provided the QoS they are authorized to receive (either through packet marking, or through routing and queueing the packets as a specific QoS assured flow). The CMTS requires authorization from a CMS (on a call-by-call basis for the telephony service) before providing access to enhanced QoS for an end-to-end IP flow. Thus, the CMTS is able to ensure that enhanced QoS is only provided for end-to-end flows that have been authorized and for which usage accounting is being done. Since the CMTS knows about the resource usage associated with individual IP flows, it generates the usage events that allow a user to be charged for service[3].

D-QoS [4] introduces the concept of a "gate" in the CMTS, which manages access to enhanced quality of service. The gate is a packet classifier and policer that ensures that only those IP flows that have been authorized by the CMS are granted access to enhanced QoS in the access and backbone networks. Gates are "admitted" selectively for a flow. For the telephony service, they are opened for individual calls. Admitting a gate involves an admission control check that is performed when a resource management request is received from the endpoint for an individual call, and it may involve resource reservation in the network for the call if necessary. The packet filter in the gate allows a flow of packets to receive enhanced QoS for a call from a specific IP source address and port number to a specific IP destination address and port number.

The gate controller is responsible for the policy decision regarding whether the gate should be opened. The Gate Controller sets up a gate in advance of a resource management message. This allows the call establishment function, which is at the gate controller, to be "stateless" in that it does not need to know the state of calls that are already in progress.

CMSs implement a set of service-specific control functions required to support the telephony service:

1) Authentication and authorization: Since services are only provided to authorized subscribers, CMSs authenticate signaling messages and authorize requests for service on a call-by-call basis.

2) Name/number translation and call routing: CMSs translate dialed E.164 numbers, or names, to a terminating IP address based on call routing logic to support a wide range of call features.

3) Service-specific admission control: CMSs can implement a broad range of admission control policies for the telephony service. For example, CMSs may provide precedence for particular calls (e.g., 911 calls). Admission control may also be used to implement overload control mechanisms, e.g. to restrict the number of calls to a particular location or to restrict the frequency of call setup to avoid signaling overload.

4) Signaling and service feature support: While many service features are implemented by intelligent endpoints, the CMS also plays a role in feature support. DCS signaling provides a set of service primitives to end-points that are mediated by the CMS/Proxy. The CMS/Proxy is involved in implementing service features that depend on the privacy of calling information, e.g., caller-ID blocking. It also plays a role in supporting service features that require users to receive a consistent view of feature operation even when an endpoint is down. For example, while an endpoint may normally participate in call forwarding, the CMS/Proxy can control call forwarding on behalf of an endpoint when the endpoint is down.

CMSs are typically organized in domains, similar to H.323 zones. A CMS is responsible for a set of endpoints and the associated CMTSs. While endpoints are not trusted, there is a trust relationship between the CMTS and its associated CMS, since the CMS plays a role as a policy server controlling when the CMTS can provide enhanced QoS service. There is also a trust relationship among CMSs. Details of the security model and mechanisms are specified in [2].

When supporting intelligent endpoints, the CMS/Proxy is designed as a simple transaction server, so that failure of a CMS/Proxy does not affect calls in progress. A domain will likely have both a primary and a secondary CMS/Proxy. If the primary CMS/Proxy fails, only calls in a transient state are affected. The endpoints involved in those calls will time out and retry. All active calls are unaffected. This is possible because the CMS/Proxy retains no call state for stable calls. We believe this design makes the CMS/Proxy efficient and highly scalable, and keeps the reliability requirements manageable.

DCS supports inter-working with the circuit switched telephone network through PSTN gateways. A PSTN gateway may be realized as a combination of a CMS/Agent, media gateway, and a signaling gateway. A media gateway acts as the IP peer of an endpoint for media packets, converting between the data format used over the IP network and the PCM format required for transmission over the PSTN. The signaling gateway acts as the IP peer of an endpoint for signaling packets, providing signaling inter-working between DCS and conventional telephony signaling protocols such as ISUP/SS7. A media gateway control protocol is used to control the operation of the media gateway from the CMS/Agent.

There are additional system elements that may be involved in providing the telephony service[1]. For example, the CMS may interface with other servers that implement the authorization or translation functions. Similarly, announcements, voicemail, and three way calling may be supported using media servers in the network. Management of security interfaces between system elements is explained in [2].

2.3 Telephony Features Supported in DCS

This specification focuses specifically on support for the following telephony features:

- Basic service features:
  - Support for voice, fax, and analog modems
  - Operator Services (0, 0+, 00, etc.)
  - E911 Emergency Services
  - Operator Break-In
  - Lawfully Authorized Electronic Surveillance
  - Terminating Announcements

- CLASS and Custom Calling Features with both flat-rate or per-use billing:
  - Call Waiting

- Caller ID/Calling Name Delivery
- Call Forwarding (no-answer, busy, all calls)
- Three-Way Calling
- Return Call
- Call Transfer

In addition to these service features, DCS is extensible to support a broad range of advanced service features (e.g., 800-number load balancing) and advanced billing models (e.g., debit cards).

We believe that the protocol framework described in this specification may be used to create a plethora of additional, new and novel features. Some examples of such additonal features that may be enabled with DCS are listed in Appendix T.

DCS extends the concept of Caller Privacy to include the ability to maintain IP addresses private. Privacy is accomplished through the cooperation of the CMSs and an application-aware (awareness of SIP) anonymizer. When the INVITE includes a Dcs-Anonymity header, the CMS sets up the appropriate translations in the anonymizer so that information such as “Caller-Num”, “Caller-Name”, or “IPAddr” are not revealed to the other party. All the messages that are transmitted hop-by-hop through the proxies are suitably modified by the proxy. End-to-end messages are routed via the anonymizers. The anonymizer translates those pieces of information in the SIP headers and SDP fields that the anonymizer is aware of that require translation.

However, true end-end privacy is maintained only if the communicating parties do not reveal their identity (in terms of “Caller-Num”, “Caller-Name”, or “IPAddr”) inadvertently in additional fields or payload of messages that they exchange end-end. In addition, we require that certain SIP headers that have the potential to reveal privacy information not be included in the end-end INVITE messages. These headers do not serve a useful purpose within these end-end INVITE messages. Therefore, DCS profiles the use of the “Contact” header, and the DCS-specific “Dcs-Remote-Party-ID” by specifying that they not be used in the end-end INVITE messages.

2.4 DCS Trust Model

DCS defines a trust boundary around the various systems and servers that are owned, operated by, and/or controlled by the service provider. These trusted systems include the proxies (also called Call Management Systems, or CMSs), the CMTSs of the cable access network, and various servers such as bridge servers, voicemail servers, announcement servers, etc. Outside of the trust boundary lie the customer premesis equipment, i.e. the MTAs, the Public Switched Telephone Network (PSTN), and various media servers operated by third-party service providers. At the boundary of the trusted domain are CMTS/Edge Routers that enforce the Quality-of-Service policies of the service provider.

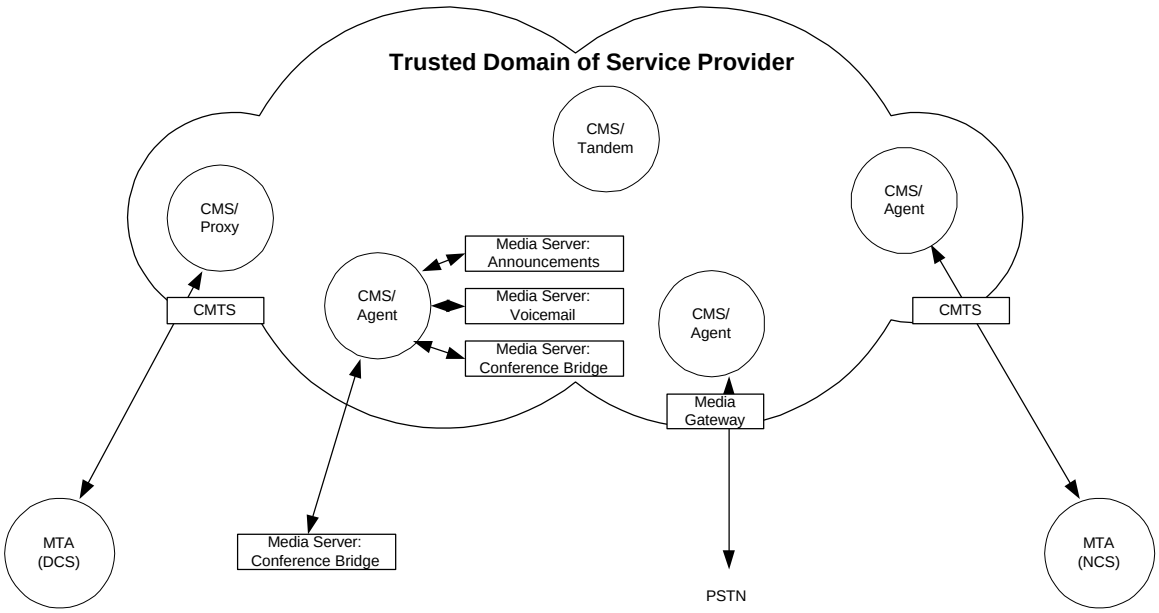


Figure 2: Trusted Domain of PacketCable Service Provider

2.5 Basic Call Flow

Figure 3 presents a high-level overview of a basic call that uses the Distributed Call Signaling specification. Figure 3 shows the most general case, where the endpoints are outside the trust boundary of the service provider, and participate directly in the SIP requests and responses. However, the same sequence of messages occur for endpoints that are within the trust boundary of the service provider, or that utilize network-based call signaling, trunk gateway control signaling, announcement controller signaling, or other signaling method between the endpoint and the CMS. In the latter cases, Figure 3 is modified to combine the vertical lines marked MTA_O and $CMS/Proxy_O$ into a single line $CMS/Agent_O$, or to combine the vertical lines marked MTA_T and $CMS/Proxy_T$ into a single line $CMS/Agent_T$, or both. Details for all four possible configurations are given in Appendix B through Appendix E. Figure 3 shows the case where the endpoints are intelligent and are capable of signaling the full-range of primitives in this specification. The description here illustrates the use of intelligent end-points and call-stateless CMS/Proxies (which are defined in subsequent sections of this specification.)

When a user goes off-hook and dials a telephone number, the originating MTA (MTA_O) collects the dialed digits and sends the initial call message, called an INVITE in SIP, to the "originating" CMS/Proxy ($CMS/Proxy_O$). This INVITE message carries with it, a Session Description, using the Session Description Protocol (SDP). Typically, when DCS endpoints or CMSs wish to ensure that adequate resources are available in the network before actual users who wish to communicate are alerted, they include additional information in the SDP of the INVITE. This additional information is a “Pre-Condition” (possibly a set of pre-conditions including both security and resource requirements that need to be met) that needs to be satisfied before the end-users are notified that they may now communicate. CMS_O verifies that MTA_O is a valid subscriber of the telephony service (using authentication information in the INVITE message) and determines whether this subscriber is authorized to place this call. CMS_O then translates the dialed number into the address of a "terminating" CMS (CMS_T) and forwards the INVITE message to it.

We assume that the originating and terminating CMSs trust each other. CMS_O augments the INVITE message that it forwards with additional information, such as billing information containing the account number of the caller. CMS_T then translates the dialed number into the address of the terminating MTA

(MTA_T) and forwards the INVITE message to MTA_T notifying it about the incoming call. The initial INVITE message invokes call feature handling at the destination MTA, such as call-forwarding. Assuming that the call is not forwarded, MTA_T negotiates the coding style and bandwidth requirements for the media streams. The 183 Session Progress response to the initial INVITE is forwarded back through the CMSs. In the figure, MTA_T sends a 183 Session Progress message to CMS/$Proxy_T$. The 183 Session Progress contains a subset of the capabilities in the INVITE message that are acceptable to MTA_T. Thus, the 183 Session Progress response includes an SDP, including the indication that the terminating MTA agrees to meet the preconditions specified in the INVITE before alerting the user. CMS_T sends a GATE-SETUP message to the terminating CMTS ($CMTS_T$) to indicate that it can *open* a gate for the IP flow associated with the phone call (the GATE-SETUP message to the terminating CMTS ($CMTS_T$), conveys policy instructions allowing $CMTS_T$ to *open* a gate for the IP flow associated with this phone call subsequent to admission control that is performed on a resource reservation request. The GATE-SETUP message may include billing information containing the account number of the subscriber that will pay for the call. This messaging between the DCS Proxy/Gate Controller and the CMTS is described in detail in the DqoS specification.). CMS_T forwards the 183 Session Progress to CMS_O. CMS_O sends a GATE-SETUP message to the originating CMTS ($CMTS_O$) to indicate that it can *open* a gate for the IP flow associated with the phone call. Finally, CMS/$Proxy_O$ forwards 183 Session Progress to MTA_O. The initial INVITE request and 183 Session Progress response contain a SIP Contact header to indicate the IP address of the remote MTA to be used for subsequent end-to-end SIP signaling exchanges. MTA_O acknowledges the 183 Session Progress directly to MTA_T using the Provisional Reliable Ack (PRACK) message. The terminating MTA_T acknowledges the PRACK message with a 200 OK message. At this point, no resources have yet been reserved, and thus the preconditions have not yet been met.

Once the initial INVITE/183 Session Progress /PRACK exchange has completed, both MTAs may request reservation of the resources that will be needed for the media streams. Once MTA_O has successfully made its reservation, it sends a PRECONDITION-MET message to MTA_T implying a command to alert the far-end user (ring the destination telephone). If MTA_T successfully reserved the resources needed for the call, it responds with both a 200 OK message acknowledging the PRECONDITION-MET message that was received as well as a 180 Ringing message to indicate that the terminating phone is ringing, and that the calling party should be given a ringback call progress tone. The originating MTA_O responds with another Provisional ACK (PRACK) to acknowledge receipt of the 180 Ringing message, and generates a local ringback call progress tone on the originating end. The terminating MTA_T responds to the PRACK with a 200 OK to acknowlege the PRACK. When the called party answers, by going off-hook, MTA_T sends a 200-OK final response that flows through the proxies, which MTA_O acknowledges. At this point the resources that were previously reserved are committed to this conversation, and the call is “cut through.”

Either party can terminate the call. An MTA that detects an on-hook sends a SIP BYE message to the remote MTA, which is acknowledged.

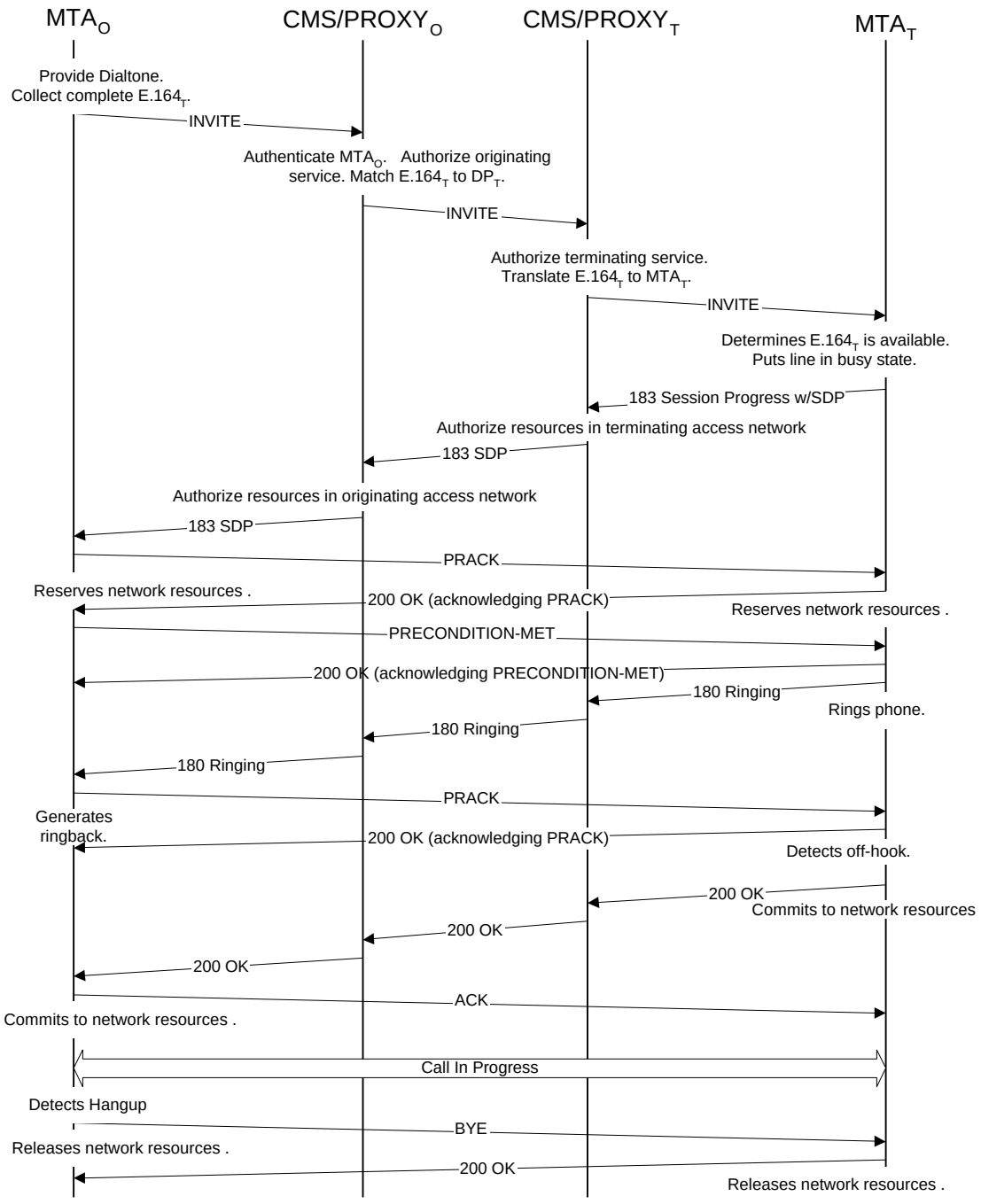


Figure 3: Ordering and synchronization of Signaling and Resource Control Protocols

3. SIP Extensions

SIP [11] has a flexible mechanism for adding extensions and new fields to the protocol for support of additional capabilities. This section defines a set of SIP extensions that enable PacketCable DCS-compliant systems to provide a robust telephone service supporting basic, CLASS, and custom calling features, while at the same time allowing the supported services to evolve.

The term UA in this section refers to an originator/destination of SIP requests, and may be either within the trust boundary or outside the trust boundary. An untrusted UA is always associated with a proxy within the boundary, who does the various value-added functions of the service provider. A trusted UA performs these value-added functions itself. An example of an untrusted UA is an intelligent MTA, which initiates SIP requests itself and sends them to a CMS/Proxy within the trust boundary. An example of a trusted UA is a NCS Call Agent, who uses the Media Gateway Control Protocol to control an endpoint outside the trust boundary, and initiates SIP requests on behalf of that endpoint. The combination of an untrusted UA with its CMS/Proxy is in many ways equivalent to a CMS/Agent; likewise a CMS/Agent may be decomposed into a UA and a Proxy (with a hidden and untestable interface between them).

This section follows the naming strategy of SIP[11], of User Agents, Clients, Servers, and Proxies. Clients initiate sessions (i.e. the call originators), and Servers accept session requests (i.e. the call terminations). A typical PacketCable MTA consists of both a Client and a Server, as it can both initiate and receive calls. The correspondence between these names and the element names in other sections of this specification is as follows:

SIP element	PacketCable element
Untrusted User Agent (UAC/UAS)	MTA (DCS)
Trusted User Agent (UAC/UAS)	CMS/Agent
Proxy	CMS/Proxy

The description of each extension in this section gives the specific procedures for untrusted UAC/UASs, for trusted UAC/UASs, and for proxies associated with the untrusted UAC/UASs.

The Distributed Call Signaling (DCS) specification extends SIP in several ways, which are summarized as follows:

- DCS supports a reservation scheme in which network resources are reserved prior to alerting the user. This is done through specification of preconditions that must be met prior to continuing the session establishment. Confirmation that the preconditions are met is indicated by an additional end-to-end message exchange, the Precondition-Met/200-OK, which occurs during the normal INVITE/200-OK/ACK exchange. This extension allows network resources to be reserved prior to alerting the user and allows network resource to be committed after the user has answered the invitation. This extension is further described in [22].

- DCS extends SIP in support of subscriber privacy, enabling subscribers to make connections without identifying themselves or revealing location information. When anonymity is not requested by the originator, calling number delivery and calling name delivery is provided to the destination (i.e. Caller-ID service) in a reliable manner. Endpoint identity is also provided to support regulatory features such as Customer Originated Trace, enabling a called party to report a harassing call even if the caller requested anonymity. This extension is further described in [20].

- DCS extends SIP with a mechanism that allows the proxies to store their state information (whatever is needed by the proxy for the duration of a call) in the endpoints, thereby reducing the storage requirements in the proxy and reducing the reliability requirements. This extension is further described in [23].

- DCS extends SIP with a media authorization mechanism, which allows the proxy to control the allocation of access network resources, and to limit the availability of the "high qos" connections and prevent theft of service. This extension is further described in [21].

- DCS extends SIP with mechanisms to pass additional information between the proxies to perform service-provider functions, such as accounting, authorization, billing, coordination of resources, electronic surveillance, etc. This extension is further described in [24].

There are several pending changes to SIP, which are used and are therefore included in this specification. These appear in current Internet-Drafts, are being actively pursued, and will likely become part of the next release of SIP (current draft is [27]). These include:

- Ability to reliably send a provisional response to a SIP request, insuring the delivery of the provisional response to the initiating UA, with retransmissions as needed. This extension is further described in [25].

- Ability to establish an early media path, from the UAS to the UAC, for purposes such as ringback and announcements played on the PSTN. This extension is further described in [16].

The remainder of this section defines the extensions to SIP REQUIRED by a DCS-compliant application. It is hoped that future versions of SIP will incorporate most, if not all, of these extensions, and that these subsections will not be necessary in future versions of DCS. As such, the Require and Proxy-Require headers are not shown in the message format requirements or call flow examples.

This section, and section 4 following, define the nearly complete set of enhancements and restrictions to a standard SIP implementation based on RFC2543[11]/RFC2543bis[27]. However, not all details of the required behavior can be captured in these sections. Later sections provide details needed for certification and interoperability testing, which are generally not present in RFC2543. Sections 3 through 11 are considered normative. Appendicies are provided to give informative examples of the use of SIP in achieving the services listed in Section 2.

3.1 SIP URL Extensions

DCS defines extensions to the SIP-URL; this specification refers to such as a DCS-URL. The DCS-URL is syntactically compatible to the SIP-URL defined in [11] section 2, Figure 3. In this specification, these extensions are used in the Request-URI, in the Dcs-Also header, in the Dcs-Remote-Party-ID header, and in the Contact header of a 3xx-Redirect response.

Further information on this extension is contained in [24].

3.1.1 Syntax

The DCS-URL is extended to allow a phone number to contain augmented information, which may include the local-number-portability office code. DCS incorporates the telephone-subscriber syntax defined in [18], which allows augmented information to be exchanged between proxies. To semantically distinguish this form of host, the token "user=phone" is included, as in [11]. The syntax description of SIP Figure 4 is extended, to be:

```
global-phone-number  = "+" 1*phonedigit [isdn-subaddress] [post-dial]
                            [area-specifier] [lrn-specifier]
local-phone-number   = 1*(phonedigit | dtmf-digit | pause-character)
                            [isdn-subsaddress] [post-dial] [area-specifier]
                            [lrn-specifier]
lrn-specifier        = "," lrn-tag "=" lrn-ident
lrn-tag              = "lrn"
lrn-ident            = token
area-specifier       = "," phone-context-tag "=" phone-context-ident
phone-context-tag    = "phone-context"
phone-context-ident  = network-prefix | private-prefix
network-prefix       = global-network-prefix | local-network-prefix
global-network-prefix = "+" 1*phonedigit
local-network-prefix = 1*(phonedigit | dtmf-digit)
private-prefix       = token
```

This specification defines an additional url-parameter, "private", to indicate that the user part of the addr-spec is in a non-intelligible form. This is used in support of endpoint privacy, and is described in [20]. The syntax description of SIP Figure 3 is extended[1] to include:

```
url-parameter        = transport-param | user-param | method-param | ttl-param |
                            maddr-param | private-param | other-param
private-param        = "private"
user-param           = "user" "=" ("phone" | "ip" | "
```

An example of a DCS-URL that includes local-number-portability information is:

sip:+1-212-555-1212,lrn=212-234@dcs-proxy.provider;user=np-queried

An example of a DCS-URL that includes private-param is:

sip: alkjsdfaoiuewrjlkdsnldbjkofsadouiewrajdf@dcs-proxy;private

3.1.2 Procedures at an Untrusted User Agent Client (UAC)

An untrusted UAC MUST NOT use the DCS-URL syntax in a request sent to other than its proxy. An untrusted UAC MUST NOT use the DCS-URL syntax other than in the Request-URI or the Dcs-Also headers.

The lrn-specifier and user-param "user=np-queried" MUST NOT appear in any DCS-URL sent by an untrusted UAC.

3.1.3 Procedures at a Trusted User Agent Client (UAC)

A DCS-URL containing the private-param MUST NOT appear in any request sent by a trusted UAC.

A DCS-URL containing the lrn-specifier and the user-param "user=np-queried" MUST NOT appear other than in an initial INVITE Request-URI sent to a proxy or trusted UAS. A trusted UAC that performs the local-number-portability lookup and passes the initial INVITE request to a proxy or trusted UAS MUST generate a Request-URI containing a DCS-URL with the user-param "user=np-queried." A trusted UAC

[1] The additional user-param value "user=np-queried" is given in [27], but does not appear in Figure 3 of that draft.

that performs the local-number-portability lookup and passes the initial INVITE request to a proxy or trusted UAS MUST include the lrn-tag indicating the returned value if the local-number-portability lookup returned a value.

3.1.4 Procedures at an Untrusted User Agent Server (UAS)

The UAS MUST NOT use the DCS-URL syntax in any responses.

A DCS-URL containing the private-param MUST be accepted in a Dcs-Also header and in a Dcs-Remote-Party-ID header. These values of DCS-URL MAY be used by the UA (in its role as a UAC) in the Request-URI of future INVITE requests.

3.1.5 Procedures at a Trusted User Agent Server (UAS)

A DCS-URL containing the private-param MUST NOT appear in any response sent by a trusted UAS.

3.1.6 Procedures at Proxy

The url-parameter “private” MUST NOT appear other than in the Request-URI for INVITE requests, or in the URL of Dcs-Also headers within an INVITE request, or in the Dcs-Remote-Party-ID header of a response, or in the Contact header of a 3xx response.

If the url-parameter “private” appears in a Request-URI or in the URL of a Dcs-Also header in an INVITE request, the proxy MUST decrypt it.

A DCS-URL containing the lrn-specifier and the user-param “user=np-queried” MUST NOT appear other than in an initial INVITE Request-URI sent to another proxy or trusted UAS. A proxy that performs the local-number-portability lookup and passes the initial INVITE request to another proxy or trusted UAS MUST generate a Request-URI containing a DCS-URL with the user-param “user=np-queried.” A proxy that performs the local-number-portability lookup and passes the initial INVITE request to another proxy or trusted UAS MUST include the lrn-tag indicating the returned value if the local-number-portability lookup returned a value.

The url-parameter “private” MUST appear in a Dcs-Remote-Party-ID header given to an untrusted endpoint when Dcs-Anonymity, as requested by the remote party, is either Full or URL. The url-parameter “private’ MUST appear in a Dcs-Remote-Party-ID header given to an untrusted endpoint when the receiving party has not subscribed to calling number delivery service. Procedures for generating a private-URL for Dcs-Remote-Party-ID are given in 3.3.1.6.2.

The url-parameter “private” MUST appear in a Dcs-Also header generated by a proxy when given to an untrusted endpoint. Procedures for generating a private-URL for Dcs-Also are given in 3.3.7.6.2.

The url-parameter “private” MUST appear in a Contact header of a 3xx response when given to an untrusted endpoint. Procedures for generating a private-URL for Contact are given in 4.6.2.

3.2 SIP Method Extensions

This section describes additional SIP request methods for support of telephone services.

3.2.1 PRACK

The PRACK method provides a simple extension to SIP for ensuring that provisional responses to all SIP requests are delivered reliably end to end, independent of the underlying transport mechanism. The extension works for provisional responses for any method. The extension is simple, requiring two new header fields, and one new method. The extension does not require support in proxies. The extension is indicated with the option tag org.ietf.sip.100rel. Further information about this extension is contained in [25]

The reliability mechanism is based on the standard windowed acknowledgement technique. When a server generates a provisional response which is to be delivered reliably, it places a sequence number (via the RSeq header field) in the provisional response. These sequence numbers are chosen with a random initial value, for security reasons. The provisional response is then retransmitted with an exponential backoff, in a fashion that is identical to final responses to INVITE. Note that a UAS does not send a response reliably unless there was a Supported header in the request indicating support for this extension[26], [27].

The reliability provided is end-to-end. Proxies do not retransmit the provisional responses; they are simply forwarded. This is similar to the way in which 200 responses for INVITE messages are handled in proxies. Note, however, that the PRACK message described here is sent reliably using the same hop-by-hop techniques for all non-INVITE requests.

The provisional response is then received at the UAC. The UAC can determine that the response is to be transmitted reliably by the presence of the RSeq header. Responses which are not transmitted reliably do not contain the RSeq header.

For a provisional response which is to be sent reliably, the UAC creates a new request, with a method of PRACK, used to acknowledge one or more provisional responses (PRACK is a cumulative acknowledgement). The PRACK request is like any other non-INVITE request sent within a call. The PRACK request contains the same Call-ID as the provisional response it is acknowledging. The CSeq number in the PRACK is higher than that of the request whose provisional response it acknowledges. The PRACK also contains a header, called RAck, which contains the highest value of the RSeq among the provisional responses being acknowledged. The RAck header also contains the contents of the CSeq field in the response being acknowledged. The combination of Call-ID, CSeq, and RAck allow the PRACK request to be matched to a set of provisonal responses within a specific transaction within a specific call. Like any other non- INVITE request, the PRACK request is retransmitted periodically up to a maximum of a four second interval. Note that the PRACK request SHOULD NOT be retransmitted when retransmissions of the provisional response are received.

When the UAS receives the PRACK request, it knows that the set of provisional responses have been received. The UAS then ceases retransmission of those provisional responses. It also generates a 200 OK response to the PRACK, and sends it to the UAC. As with any other non-INVITE request, the 200 response to the PRACK request MUST be retransmitted when retransmissions of the PRACK request are received.

When the UAC receives the 200 response (or any other final response) to the PRACK, it stops retransmitting the PRACK. This is standard behavior for non-INVITE requests.

PRACK requests MAY contain bodies. This is useful for establishing early media sessions for tones and announcements, or for setting up security or network preconditions for call completion (see section 3.2.2).

3.2.1.1 Message Format

The PRACK message headers and contents is described as follows:

Header:	***Requirement for contents***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header received in the message being acknowledged*
Via:	*MUST be present, as in a normal SIP request message.*
From:	*MUST be present.* *MUST be copies of same headers in the provisional response.*
To:	
Call-ID:	
Cseq: n_o+1 PRACK	*MUST be present.* *Sequence number 'n_o+1' MUST be one higher than previous sequence number used within this call leg, method MUST indicate PRACK*
Rack: x n_o <Method>	*MUST be present.* *Value 'x' MUST be a copy of the value in the Rseq header of the provisional response being acknowledged. Value 'n_o' MUST be a copy of the Cseq value from the original request. Method MUST be of the original request, typially INVITE.*
Content-Type:	*MUST be present if body is present.*
Content-length:	*MUST be present if body is present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
<Message body>	*Message body MAY be present.*

3.2.1.2 Procedures at an Untrusted User Agent Client (UAC)

The UAC MUST include a Supported header with the name org.ietf.sip.100rel listed as a feature token. The request whose provisional response is being reliably sent is referred to as the initial request.

If a provisional response is received for the initial request, and that response contains an RSeq header, the response is to be sent reliably. If the response is a 100 (as opposed to 101 to 199), the RSeq in the response is ignored. The reliability mechanisms defined here MUST NOT be used on 100 responses.

If the received provisional response was not a 100, and contained an RSeq header, the UAC MUST create a new request with method PRACK. The Call-ID in this request MUST match that of the provisional response. The CSeq in this request MUST be larger than the last request (PRACK or otherwise) sent by this UAC for this call leg. The To, From, and Via headers MUST be present, and MUST be constructed as they would be for a re-INVITE or BYE as specified in [11]. In particular, if the provisional response contained a tag in the To field, this tag MUST be mirrored in the To field of the PRACK.

Since reliable provisional responses MAY contain Record-Route and Contact headers, the PRACK request MUST contain Route headers if the Record-Route headers were present in the provisional response. The Request-URI and Route header are constructed as specified in [11]. The Route header that is constructed from some provisional response MUST NOT be placed in any other request except for the PRACK for that provisional response.

A UAC MUST NOT insert a Route header into a PRACK request if no Record-Route header was present in the response.

PRACK requests MAY contain bodies. This is useful for establishing early media sessions for tones and announcements, or for setting up security or network preconditions for call completion.

Once the PRACK request is created, it is sent by the UAC. It is sent as would any other non-INVITE request for a call, using the same retransmission strategy as other requests. Note that a UAC SHOULD NOT retransmit the PRACK request when it receives a retransmission of the provisional response being

acknowledged, although doing so does not create a protocol error. As with any other non-INVITE request, the UAC continues to retransmit the PRACK request until it receives a final response. A reliable provisional response for which a PRACK request has been sent is called an acknowledged reliable provisional response.

Handling of subsequent reliable provisional responses for the same request follows the same rules as above, with the following difference. Reliable provisional responses are guaranteed to be in order. As a result, if the UAC receives a reliable provisional response, and its RSeq value isn't one higher than the previous acknowledged reliable provisional response, that response MUST NOT be acknowledged with a PRACK. An implementation MAY discard the response, or MAY cache the response in the hopes of receiving the missing responses. Note that this requires the UAC to store the RSeq value of the last acknowledged reliable provisional response for the duration of the transaction.

3.2.1.3 Procedures at a Trusted User Agent Client (UAC)

The procedures at a trusted UAC are identical to those of an untrusted UAC, as given in 3.2.1.2.

3.2.1.4 Procedures at an Untrusted User Agent Server (UAS)

The UAS MAY send any provisional response reliably, so long as the initial request contained a Supported header indicating that this feature is understood. The UAS MUST NOT attempt to send a 100 response reliably. Only responses numbered 101 to 199 MAY be sent reliably. The rest of this discussion assumes that the initial request contained a Supported header listing this feature, and that there is a response to be sent reliably.

The provisional response to be sent reliably MUST include an RSeq header. The numeric value of this header is chosen randomly for the first provisional response for a given request. The value in each subsequent reliable provisional response for the same request MUST be greater by exactly one. The RSeq numbering space is within a single request. This means that provisional responses for different requests MAY use the same values for the RSeq number.

Reliable provisional respones MAY contain a body. As with any other response, reliable provisional responses MUST mirror the From, Call-ID, CSeq, Via, and To fields from the request. The UAS MUST insert a tag into the To field of the provisional response. The reliable provisional response MUST contain a Contact header.

The reliable provisional response is retransmitted periodically, even if sent over TCP. The retransmission strategy is the same as for a final response to an INVITE (see section 7.2). If no PRACK is received for that response after 96 seconds, it is considered a network or endpoint failure. Behavior at that point is at the discretion of the implementor.

The UAS then waits for a PRACK request. It matches the PRACK request to a reliable provisional response through the Call-ID, To, and From, which identify the call-leg of the PRACK, and through the RAck header, which identifies the particular request and provisional response within the call leg. Specifically, a PRACK request X matches a provisional response Y if all of the following are true:

- The Call-ID in X matches the Call-ID in Y.
- The From in X matches the From in Y, including the tag, if present.
- The To in X matches the To in Y, including the tag, if present. If Y did not contain a tag, but X did, these do not match. If Y did contain a tag, but X does not, these do match.
- The method in the RAck of X matches the method in the CSeq of Y.
- The CSeq-num in the RAck matches the CSeq number in Y.
- The response-num in the RAck is greater than or equal to the RSeq value in Y.

Note that a single PRACK may match multiple provisional responses. Only one response is sent to the PRACK.

If a PRACK request is received that does not match any reliable provisional response, the UAS responds to the PRACK with a 481 response.

If a PRACK request is received that does match some provisional responses for which no PRACK has been received, the provisional response retransmissions for those responses cease. The UAS generates a 200 OK response to the PRACK, and sends it. The rules for generation of the 200 OK for the PRACK, and for its transmission, follow those for any non-INVITE method. The UAS can be certain at this point that those provisional responses have been received in order.

If a PRACK request is received that does match some provisional responses, but a different PRACK has been received for all those responses already (different meaning the PRACK had a different CSeq value), the new PRACK is responded to with a 200 OK. There is no need to stop retransmissions of those reliable provisional responses that match, since their retransmissions will have already ceased from the previous PRACK.

If the PRACK contained a body, the body is treated in the same way a body in an ACK is treated.

As with any other non-INVITE request, if a retransmission of the PRACK request is received, the response to the PRACK is retransmitted. There is no need to retransmit the reliable provisional response when a PRACK is received.

After the first reliable provisional response for a request has been acknowledged, the UAS MAY send additional reliable provisional responses. The UAS MUST NOT send a second reliable provisional response until the first is acknowledged. After the first, it is RECOMMENDED that the UAS not send additional reliable provisional responses until the previous is acknowledged. The first reliable provisional response receives special treatment because it conveys the intitial sequence number. If additional reliable provisional responses were sent before the first is acknowledged, the UAS could not be certain these were received in order.

3.2.1.5 Procedures at a Trust User Agent Server (UAS)

The procedures at a trusted UAS are identical to those of an untrusted UAS, as given in 3.2.1.4.

3.2.1.6 Procedures at Proxy

This extension does not require active participation from proxies. As far as they are concerned, the PRACK is just another request to be forwarded. In most cases, the PRACK will be sent directly end-to-end, avoiding the proxies.

The only requirement for proxies is that they MUST pass all provisional responses, as mandated in [27]

3.2.2 PRECONDITION-MET

This section discusses how network QoS and security establishment can be made a precondition to sessions initiated by the Session Initiation Protocol (SIP)[11], and described by SDP [12]. These preconditions require that the participant reserve network resources (or establish a secure media channel) before continuing with the session. We do not define new QoS reservation or security mechanisms; these pre-conditions simply require a participant to use existing resource reservation and security mechanisms before beginning the session. This extension is further described in [22].

This results in a multi-phase call-setup mechanism, with the resource management protocol interleaved between two phases of call signaling. The objective of such a mechanism is to enable deployment of robust IP Telephony services, by ensuring that resources are made available before the phone rings and the participants of the call are "invited" to participate.

The general idea behind the extension is simple. We define two new SDP attributes, "X-pc-qos" and "X-pc-security". The "qos" attribute indicates whether end-to-end resource reservation is optional or mandatory, and in which direction (send, recv, or sendrecv). When the attribute indicates mandatory, this means that the participant who has received the SDP MUST NOT proceed with participation in the session until resource reservation has completed in the direction indicated. In this case, "not proceeding" means that the participant behaves as if they had not received the SDP at all. If the attribute indicates that QoS for the stream is optional, then the participant SHOULD proceed normally with the session, but SHOULD reserve network resources in the direction indicated, if they are capable. Absence of the "qos" attribute means the participant MAY reserve resources for this stream, and SHOULD proceed normally with the session. This behavior is the normal behavior for SDP.

The direction attribute indicates which direction reservations should be reserved in. If "send", it means reservations should be made in the direction of media flow from the session originator to participants. If "recv", it means reservations should be made in the direction of media flow from participants to the session originator. In the case of "sendrecv", it means reservations should be made in both directions.

Either party MAY include a "confirm" attribute in the SDP. When the "Confirm" attribute is present, the recipient MUST send a PRECONDITION-MET message to the sender, with SDP attached, telling the status of each precondition as "success" or "failure." If the "confirm" attribute is present in the SDP sent by the session originator to the participant (e.g. in the SIP INVITE message), then the participant MUST send the PRECONDITION-MET message to the originator. If the "confirm" attribute is present in the SDP sent by the recipient to the originator (e.g. in a SIP response message), then the originator MUST send the PRECONDITION-MET message to the participant.

The PRECONDITION-MET method is used for communicating successful completion of preconditions from the calling to called user agents.

The signaling path for the PRECONDITION-MET method is the signaling path established as a result of the call setup. This can be either direct signaling between the calling and called user agents or a signaling path involving SIP proxy servers that were involved in the call setup and added themselves to the Record-Route header on the initial INVITE message.

The precondition information is communicated in the message body, which MUST contain an SDP. For every agreed precondition, the strength-tag MUST indicate "success" or "failure".

The Call-ID in the PRECONDITION-MET MUST match that of the response. The CSeq in this request MUST be larger than the last request sent by this UAC for this call leg. The To, From, and Via headers MUST be present, and MUST be constructed as they would be for a re-INVITE or BYE as specified in [11]. In particular, if the provisional response contained a tag in the To field, this tag MUST be mirrored in the To field of the PRECONDITION-MET.

Once the PRECONDITION-MET request is created, it is sent by the UAC. It is sent as would any other non-INVITE request for a call. In particular, when sent over UDP, the PRECONDITION-MET request is retransmitted with an exponentially increasing interval (see section 7.2). As with any other non-INVITE request, the UAC continues to retransmit the PRECONDITION-MET request until it receives a final response.

The session originator (UAC) prepares an SDP message body for the INVITE describing the desired QoS and security preconditions for each media flow, and the desired directions. The token value "send" means the direction of media from originator (whichever entity created the SDP) to recipient (whichever entity

received the SDP in a SIP message), and "recv" is from recipient to originator. In an INVITE, the UAC is the originator, and the UAS is the recipient. The roles are reversed in the response.

The recipient of the INVITE (UAS) returns a 183-Session-Progress provisional response containing SDP, along with the qos/security attribute for each stream having a precondition, and would typically include a confirmation request. This SDP is a subset of the preconditions indicated in the INVITE. Unlike normal SIP processing, the UAS MUST NOT alert the called user at this point. The UAS now attempts to reserve the qos resources and establish the security associations.

The 183-Session-Progress is received by the UAC. If the 183 contained SDP with mandatory qos/security parameters, the UAC SHOULD NOT generate local ringback until the mandatory preconditions are met. The UAC attempts to reserve the needed resources and establish the security associations.

If either party requests a confirmation, a PRECONDITION-MET message MUST be sent to that party. The PRECONDITION-MET message contains the success/failure indication for each precondition. Upon receipt of the PRECONDITION-MET message, the UAC/UAS continues normal SIP call handling, by (for a UAS) alerting the user and sending either a 180-Ringing or 183-Early-Media provisional response. The UAC either provides ringback (in the case that a 180 was received) or plays media from the remote party (in the case of 183), and the SIP transaction completes normally.

Note that this extension requires usage of reliable provisional responses, as described in Section 3.2.1. This is because the 183 contains SDP with information required for the session originator to initiate reservations from it towards the participant.

3.2.2.1 Message Format

The PRECONDITION-MET message headers and contents is described as follows:

Header:	***Requirement for contents***
PRECONDITION-MET SIP-URL SIP/2.0	*MUST be present. Method MUST be PRECONDITION-MET. The value of the SIP-URL MUST be the Contact header received in the Initial INVITE or initial 183-Session-Progress message*
Via:	*MUST be present, as in a normal SIP request message.*
From:	*MUST be present.* *MUST be copies of same headers in the provisional response.*
To:	
Call-ID:	
Cseq: n_o+1 PRECONDITION-MET	*MUST be present.* *Sequence number 'n_o+1' MUST be one higher than previous sequence number used within this call leg, method MUST indicate PRECONDITION-MET*
Content-Type: application/sdp	*MUST be present.* *MUST indicate "application/sdp"*
Content-length:	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
<SDP description >	*Message body MUST be present.*

3.2.2.2 Procedures at an Untrusted User Agent Client (UAC)

Unless stated otherwise, the protocol rules for the PRECONDITION-MET request governing the usage of tags, Route and Record-Route, retransmission and reliability, CSeq incrementing and message formatting follow those in [11] as defined for the BYE request.

The session originator (UAC) MAY include QoS and security preconditions (including the desired direction) for each media flow in the SDP sent with the INVITE. The token value "send" means the direction of media from originator (whichever entity created the SDP) to recipient (whichever entity received the SDP in a SIP message), and "recv" is from recipient to originator.

Upon receipt of the 183-Session-Progress with SDP, the UAC MUST initiate the qos reservations and establish the security associations required.

If any of the mandatory preconditions cannot be met, the UAC MUST send a CANCEL and terminate the session.

When the optional preconditions have either been met or have failed, or the mandatory preconditions have been met, and the SDP received from the UAS included a confirmation request, the UAC MUST send a PRECONDITION-MET message to the UAS with SDP, where each precondition is updated to indicate success/failure.

The session now completes normally, as per [11].

3.2.2.3 Procedures at a Trusted User Agent Client (UAC)

Unless stated otherwise, the protocol rules for the PRECONDITION-MET request governing the usage of tags, Route and Record-Route, retransmission and reliability, CSeq incrementing and message formatting follow those in [11] as defined for the BYE request.

The session originator (UAC) MAY include QoS and security preconditions (including the desired direction) for each media flow in the SDP sent with the INVITE. The token value "send" means the direction of media from originator (whichever entity created the SDP) to recipient (whichever entity received the SDP in a SIP message), and "recv" is from recipient to originator.

Upon receipt of the 183-Session-Progress with SDP, the UAC MUST initiate the qos reservations and establish the security associations required.

If any of the mandatory preconditions cannot be met, the UAC MUST send a CANCEL and terminate the session.

When the optional preconditions have either been met or have failed, or the mandatory preconditions have been met, and the SDP received from the UAS included a confirmation request, the UAC MUST send a PRECONDITION-MET message to the UAS with SDP, where each precondition is updated to indicate success/failure.

The session now completes normally, as per [11].

3.2.2.4 Procedures at an Untrusted User Agent Server (UAS)

Unless stated otherwise, the protocol rules for the PRECONDITION-MET request governing the usage of tags, Route and Record-Route, retransmission and reliability, CSeq incrementing and message formatting follow those in [11] as defined for the BYE request.

On receipt of an INVITE request containing preconditions, the UAS MUST generate a 183-Session-Progress response containing a subset of the preconditions supported by the UAS. In the response, the token value "send" means the direction of the media from the UAS to the originator, and "recv" is from the originator to the recipient. This is reversed from the SDP in the initial INVITE. The 183 provisional response MUST include a Session: header with parameter "qos" and/or "security" and MUST NOT include the session parameter "Media."

Unlike normal SIP processing, the UAS MUST NOT alert the called user at this point (unless the SDP in the 183 indicated no mandatory preconditions and no confirmation requests). The UAS now attempts to reserve the qos resources and establish the security associations. The UAS MAY set a local timer to limit the time waiting for preconditions to complete.

If the UAS is unable to perform any mandatory precondition, it MUST send a 580-Precondition-Failure response to the UAC (see section 3.4.2).

If the UAS had requested confirmation of a precondition in the response SDP, it SHOULD wait for the PRECONDITION-MET message from the originator containing the success/failure indication of each precondition from the originator's point of view. If that confirmation indicates a failure for a mandatory precondition, the UAS MUST send a 580-Precondition-Failure response to the UAC.

Once the preconditions are met, the UAS alerts the user, and the SIP transaction completes normally.

3.2.2.5 Procedures at a Trusted User Agent Server (UAS)

Unless stated otherwise, the protocol rules for the PRECONDITION-MET request governing the usage of tags, Route and Record-Route, retransmission and reliability, CSeq incrementing and message formatting follow those in [11] as defined for the BYE request.

On receipt of an INVITE request containing preconditions, the UAS MUST generate a 183-Session-Progress response containing a subset of the preconditions supported by the UAS. In the response, the token value "send" means the direction of the media from the UAS to the originator, and "recv" is from the originator to the recipient. This is reversed from the SDP in the initial INVITE. The 183 provisional response MUST include a Session: header with parameter "qos" and/or "security" and MUST NOT include the session parameter "Media."

Unlike normal SIP processing, the UAS MUST NOT alert the called user at this point (unless the SDP in the 183 indicated no mandatory preconditions and no confirmation requests). The UAS now attempts to reserve the qos resources and establish the security associations. The UAS MAY set a local timer to limit the time waiting for preconditions to complete.

If the UAS is unable to perform any mandatory precondition, it MUST send a 580-Precondition-Failure response to the UAC (see section 3.4.2).

If the UAS had requested confirmation of a precondition in the response SDP, it SHOULD wait for the PRECONDITION-MET message from the originator containing the success/failure indication of each precondition from the originator's point of view. If that confirmation indicates a failure for a mandatory precondition, the UAS MUST send a 580-Precondition-Failure response to the UAC.

Once the preconditions are met, the UAS alerts the user, and the SIP transaction completes normally.

3.2.2.6 Procedures at Proxy

Unless stated otherwise, the protocol rules for the PRECONDITION-MET request at a proxy are identical to those for a BYE request as specified in [11].

3.3 SIP Header Extensions

This section describes extensions to SIP headers for support of telephone services. This section includes modifications to existing SIP headers as well as definitions of new SIP headers. DCS compliant devices MUST follow the syntax and usage rules of these header extensions as defined in this section.

3.3.1 DCS-REMOTE-PARTY-ID

In the telephone network, calling identity information is needed to support the Calling Number Delivery and Calling Name Delivery services which provide the called party with identity information about the calling party prior to the called party answering the call; the calling party is here identified as the station originating the call. In order for this service to be dependable, the called party must be able to trust that the calling identity information being presented is valid. Consider for example a tele-marketer presenting himself with the identity of one of your co-workers, or, even worse, an automated credit-card activation system using calling identity information as an authentication check. In order for the calling identity information to be trustworthy, the information must come from a trusted source.

Calling identity information may also be needed to support regulatory requirements for a public telephony service. An example of this is the Customer Originated Trace service, which enables a called party to have the identity of a calling party recorded by the telephony service provider. This enables, e.g., the receiver of harassing phone calls to make the identity of the originator of such calls available to the proper authority. Again, in order for this service to be useful, the Calling Identity information recorded must be trustworthy.

One scenario for establishing such trust is for an untrusted SIP user agent to require that all incoming SIP invitations arrive through a trusted SIP proxy. For simplicity we will assume that each SIP user agent is associated with a single SIP proxy. Proxies are interconnected with other proxies which may or may not trust each other. When a SIP user agent originates a call through its proxy, it trusts that the proxy will carry out the service requested, even if other proxies are involved. The proxy however does not trust the SIP user agent since it will typically reside at the customer premise.

For the Calling and Called Identity Delivery, we assume that an untrusted SIP user agent can determine if invitations are arriving through its proxy, and thereby can be trusted, or not. Furthermore, as in the current telephone network, the trusted proxy is assumed to either receive or possess calling party information that enables it to determine the identity of the calling party. In the following, we will use the term remote party to refer to calling and called party, where a distinction is not important.

When a call is placed, the calling identity delivery services reveal privacy information to the called party, and the calling party therefore has the option to block the delivery of this information to the called party. In the PSTN, this is typically achieved by subscribing to a Calling Identity Delivery Blocking service but can be done on an individual call basis as well. When the Calling Identity Delivery Blocking Service is invoked, information about the calling party is still passed through the trusted intermediaries, however presentation restriction indicators are set in the signaling messages to signal the far-end side, that the calling identity information is not to be provided to the called party.

More generally, we may say that the service provided is that of preventing the called party from obtaining information about the calling party that may either be used to identify the party or reveal location information about the party. In an IP environment, IP addressing information may provide the other party with information to reach or identify the calling party. IP addressing information may reveal some level of location information, for instance if one has knowledge of which addresses are deployed where, or by revealing that a given caller is using a different IP-address or address block than usual.

When such a privacy service is to be provided in a SIP environment, it leads to two requirements. First, calling identity information present in SIP messages must not be delivered in an intelligible form to the called party, yet it must be possible to determine the identity of the call originator even in the case where the call is routed through one or more untrusted intermediaries. Secondly, when using SIP in an IP

environment, IP addressing information must be able to be hidden from the other party. Furthermore, in an IP environment, these requirements apply equally well in the opposite direction, i.e. the calling party may wish to identify the called party and the called party may have privacy concerns as well.

Since DCS-compliant clients encrypt the SIP From header, the Dcs-Remote-Party-ID extension is added to an INVITE message to identify the caller. The Dcs-Remote-Party-ID header is inserted by the UAC, and is verified by the Proxy. The terminating Proxy forwards the Dcs-Remote-Party-ID header unchanged to the UAS only if it has subscribed to Caller ID/Calling Name service and the originator has not requested privacy. Further discussion on this extension is given in [20].

3.3.1.1 Syntax

The BNF description of this header is:

Dcs-Remote-Party-ID = "Dcs-Remote-Party-ID" ":" [display-name] "<" addr-spec ">" *(";" rpi-token)
rpi-token = rpi-id | rpi-type | rpi-auth | other-rpi-token
rpi-id = "rpi-id" "=" ("private" | "unavailable" | "na")
rpi-type = "rpi-type" "=" ("operator" | token)
rpi-auth = "auth" "=" auth-scheme #auth-param
auth-scheme = token
auth-param = token "=" (token | quoted-string)
other-rpi-token = token ["=' (token | quoted-string)]

Display-name is a text string that identifies the account name of the remote endpoint. Where privacy is requested by the remote endpoint, the display-name is deleted from the header.

Addr-spec contains a URL that can be used by the UA to identify the remote endpoint in further SIP messages. Addr-spec typically contains a tel: URL giving the identity of the remote endpoint, or a DCS-URL with a private-param when privacy is requested by the remote endpoint. At the called party, this contains the identity of the caller, and at the calling party, this contains the translated identity of the called party. The addr-spec may be a SIP-URL if the remote party is not identified by a telephone number. Where privacy is requested by the remote endpoint, the addr-spec is a DCS-URL with an encrypted string as username, and the proxy name as hostname, and a url-parameter of "private".

Rpi-id, when present, gives further information to an endpoint regarding the interpretation of the addr-spec. The string "private" means the remote party requested name privacy. The string "unavailable" means the information is not available to the proxy. The string "na" means the endpoint has not subscribed to identity delivery service.

Rpi-type identifies any special privileges given to the originator. The string "Operator" is a reserved identifier supplied by the network to the UA.

Rpi-auth is described in [20]. It is not presently used in this specification.

To maintain privacy of the initiator, an INVITE request sent on the end-end signaling path SHOULD NOT contain a Dcs-Remote-Party-ID header.

3.3.1.2 Procedures at an Untrusted User Agent Client (UAC)

The originating UAC SHOULD insert a Dcs-Remote-Party-ID header into the initial INVITE message for a new call.

The addr-spec MUST contain the pre-provisioned URL (either a sip URL or a tel URL) assigned by the service provider for the specific originating line. Note this URL is the same string as appears in the Request-URI for incoming call attempts, to identify the particular desired destination. It SHOULD be either a tel: URL or a sip: URL with user=phone.

The display-name MAY be provided by the UAC. If present, it SHOULD be one of a set of pre-provisioned strings associated with the originating line by the service provider. If not present, or if not one of the pre-provisioned strings associated with the originating line by the service provider, it will be orvewritten by the proxy.

Rpi-id MUST NOT appear in the Dcs-Remote-Party-ID header in the initial INVITE message.

Rpi-type MAY be present in the Dcs-Remote-Party-ID header of the initial INVITE message, and if present MUST be set to "operator." If the originating line is not pre-provisioned as providing operator services, the rpi-type will be deleted by the proxy.

The value of addr-spec in the Dcs-Remote-Party-ID header in the response to the INVITE is used in initiating certain call control functions, as described with Dcs-Also (section 3.3.7).

The untrusted UAC MAY use the Dcs-Remote-Party-ID header of the response to provide information about the actual destination of the call. If rpi-id=private is present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, the UAC SHOULD indicate that the caller-identification (both caller-number and calling-name) was blocked by the remote party. If rpi-id=na is present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, the UAC SHOULD indicate that neither calling-number nor calling-name service is subscribed. If rpi-id=unavailable is present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, the UAC SHOULD indicate that the caller identity (both calling-number and calling-name) is unavailable. If neither rpi_id=na nor rpi-id=unavailable is present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, the value of Dcs-Remote-Party-ID addr-spec SHOULD be displayed as the calling-number. If display-name is present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, it SHOULD be used for the calling-name delivery service. If display-name is not present in the Dcs-Remote-Party-ID header of the response to an initial INVITE, the UAC SHOULD indicate that calling-name service is not subscribed.

3.3.1.3 Procedures at a Trusted User Agent Client (UAC)

The originating UAC MUST insert a Dcs-Remote-Party-ID header into the initial INVITE message for a new call.

The addr-spec MUST contain the pre-provisioned URL (either a sip URL or a tel URL) assigned by the service provider for the specific originating line. It SHOULD be either a tel: URL or a sip: URL with user=phone.

The display-name SHOULD be provided by the UAC. If present, it MUST be one of a set of pre-provisioned strings associated with the originating line by the service provider.

Rpi-id MUST NOT appear in the Dcs-Remote-Party-ID header in the initial INVITE message.

If the originating line is pre-provisioned as providing operator services, Rpi-type MAY be present in the Dcs-Remote-Party-ID header of the initial INVITE message, and if present MUST be set to "operator."

The value of addr-spec in the Dcs-Remote-Party-ID header in the response to the INVITE is used in initiating certain call control functions, as described with Dcs-Also (section 3.3.7).

A trusted UAC MUST use the value of the Dcs-Anonymity header of the response to an initial INVITE to limit any display of the Dcs-Remote-Party-ID information. If the Dcs-Anonymity header of the response to

an initial INVITE includes URL or Full, the UAC MUST NOT reveal the value of addr-spec. If the Dcs-Anonymity header of the response to an initial INVITE includes Name or Full, the UAC MUST NOT reveal the value of display-name.

3.3.1.4 Procedures at an Untrusted User Agent Server (UAS)

The destination UAS SHOULD insert a Dcs-Remote-Party-ID header into the first non-100 response to the initial INVITE message for a new call.

The addr-spec MUST contain the pre-provisioned URL (either a sip URL or a tel URL) assigned by the service provider for the specific terminating line. Note this URL is the same string as appeared in the Request-URI for the incoming call attempt, which identified the particular desired destination.

The display-name MAY be provided by the UAS. If present, it SHOULD be one of a set of pre-provisioned strings associated with the terminating line by the service provider. If not present, or if not one of the pre-provisioned strings associated with the terminating line by the service provider, it will be orvewritten by the proxy.

Rpi-id MUST NOT appear in the Dcs-Remote-Party-ID header in the reponse to the initial INVITE message.

Rpi-type MAY be present in the Dcs-Remote-Party-ID header of the response to the initial INVITE, and if present MUST be set to “operator.” If the terminating line is not pre-provisioned as providing operator services, the rpi-type will be deleted by the proxy.

The untrusted UAS SHOULD NOT use the display-name or addr-spec of the From header of the initial INVITE to display caller-identification information, and SHOULD instead use the Dcs-Remote-Party-ID header of the initial INVITE request to provide the information. If rpi-id=private is present in the Dcs-Remote-Party-ID header of the initial INVITE, the UAS SHOULD indicate that the caller-identification (both caller-number and calling-name) was blocked by the remote party. If rpi-id=na is present in the Dcs-Remote-Party-ID header of the initial INVITE, the UAS SHOULD indicate that the service is not subscribed. If rpi-id=unavailable is present in the Dcs-Remote-Party-ID header of the initial INVITE, the UAS SHOULD indicate that the caller identity (both calling-number and calling-name) is unavailable. If neither rpi-id=na nor rpi-id=unavailable is present in the Dcs-Remote-Party-ID header of the initial INVITE, the value of Dcs-Remote-Party-ID addr-spec SHOULD be displayed as the calling-number. If display-name is present in the Dcs-Remote-Party-ID header of the initial INVITE, it SHOULD be used for the calling-name delivery service. If display-name is not present in the Dcs-Remote-Party-ID header of the initial INVITE, the UAS SHOULD indicate that calling-name service is not subscribed.

A trusted UAS MUST use the Dcs-Anonymity header value to control the display of the Dcs-Remote-Party-ID information. If Dcs-Anonymity includes URL or Full, or the subscriber has not subscribed to calling number delivery, the UAS MUST NOT reveal the value of addr-spec. If Dcs-Anonymity includes Name or Full, or the subscriber has not subscribed to calling name delivery, the UAS MUST NOT reveal the value of display-name.

3.3.1.5 Procedures at a Trusted User Agent Server (UAS)

The destination UAS MUST insert a Dcs-Remote-Party-ID header into the first non-100 response to the initial INVITE message for a new call.

The addr-spec MUST contain the pre-provisioned URL (either a sip URL or a tel URL) assigned by the service provider for the specific terminating line.

The display-name SHOULD be provided by the UAS. If present, it MUST be one of a set of pre-provisioned strings associated with the terminating line by the service provider.

Rpi-id MUST NOT appear in the Dcs-Remote-Party-ID header in the reponse to the initial INVITE message.

If the terminating line is pre-provisioned as providing operator services, Rpi-type MAY be present in the Dcs-Remote-Party-ID header of the response to the initial INVITE message, and if present MUST be set to "operator."

A trusted UAS MUST use the value of the Dcs-Anonymity header in the initial INVITE to control the display of the Dcs-Remote-Party-ID information. If the Dcs-Anonymity header of the initial INVITE includes URL or Full, the UAS MUST NOT reveal the value of addr-spec. If the Dcs-Anonymity header of the initial INVITE includes Name or Full, the UAS MUST NOT reveal the value of display-name.

3.3.1.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to an untrusted endpoint, performs both sets of procedures.

A proxy that is neither an originating proxy nor a terminating proxy MUST pass the Dcs-Remote-Party-ID header unchanged.

3.3.1.6.1 Procedures at Originating Proxy

If the Dcs-Remote-Party-ID header is not present in the initial INVITE request from the UAC, the originating proxy MUST insert one.

The Remote-Party-ID header field MUST be verified and possibly modified by it's the originating proxy. The proxy MUST verify the URL is a valid URL for the initiating line; if not the proxy MUST rewrite the URL with a valid URL for the initiating device. If the URL parameter contains a sip URL with user=phone, the proxy MUST expand the username into a full E.164 number, and replace it with a tel URL. The proxy MUST verify the display-name is a valid string for the given initiating line; if not the proxy MUST rewrite the display-name with a valid string for the line. If rpi-type is present in the Remote-Party-ID header field, the proxy MUST verify the endpoint is entitled to the special status indicated; if not the proxy MUST remove the rpi-type token from the header. If the proxy does not recognize the rpi-type token value, it MUST remove that token from the header.

If the Dcs-Anonymity header is present in the response to the initial INVITE, with value Full or URL, the proxy MUST replace the URL in the Dcs-Remote-Party-ID header in the response to the initial INVITE with a private URL and add "rpi-id=private." If the Dcs-Anonymity header is present in the response to the initial INVITE, with value Full or Name, the proxy MUST delete the display-name in the Dcs-Remote-Party-ID header. If the UAC has not subscribed to calling name delivery, then the proxy MUST delete the display-name in the Dcs-Remote-Party-ID header in the response to the initial INVITE. If the UAC has not subscribed to calling number delivery, then the proxy MUST replace the URL in the Dcs-Remote-Party-ID header in the response to the initial INVITE with a private URL and add "rpi-id=na."

To generate the username of a private URL, the proxy MUST include (1) the initial URL, (2) the value of Dcs-Anonymity, and (3) sufficient checksum information to prevent tampering by the untrusted endpoint. It MAY contain any other information the proxy desires. This information MUST be encoded or encrypted such that the endpoint is unable to discern the initial URL. It is RECOMMENDED that the string be encrypted with a symmetric privately-held key, and converted to a printable string using Base64 encoding.

The proxy MUST identify itself in the hostname of the private URL.

3.3.1.6.2 Procedures at Terminating Proxy

If the Dcs-Anonymity header is present in the initial INVITE, with value Full or URL, the proxy MUST replace the URL in the Dcs-Remote-Party-ID header in the initial INVITE with a private URL and add "rpi-id=private." If the Dcs-Anonymity header is present in the initial INVITE, with value Full or Name, the proxy MUST delete the display-name in the Dcs-Remote-Party-ID header in the initial INVITE. If the UAS has not subscribed to calling name delivery, then the proxy MUST delete the display-name in the Dcs-Remote-Party-ID header in the initial INVITE. If the UAS has not subscribed to calling number delivery, then the proxy MUST replace the URL in the Dcs-Remote-Party-ID header in the initial INVITE with a private URL and add "rpi-id=na."

To generate the username of a private URL, the proxy MUST include (1) the initial URL, (2) the value of Dcs-Anonymity, and (3) sufficient checksum information to prevent tampering by the untrusted endpoint. It MAY contain any other information the proxy desires. This information MUST be encoded or encrypted such that the endpoint is unable to discern the initial URL. It is RECOMMENDED that the string be encrypted with a symmetric privately-held key, and converted to a printable string using Base64 encoding.

The proxy MUST identify itself in the hostname of the private URL.

If the Dcs-Remote-Party-ID header is not present in the response to the initial INVITE request, from the UAS, the terminating proxy MUST insert one.

The Remote-Party-ID header field MUST be verified and possibly modified by it's the terminating proxy. The proxy MUST verify the URL is a valid URL for the terminating line; if not the proxy MUST rewrite the URL with a valid URL for the terminating device. If the URL parameter contains a sip URL with user=phone, the proxy MUST expand the username into a full E.164 number, and replace it with a tel URL. The proxy MUST verify the display-name is a valid string for the given terminating line; if not the proxy MUST rewrite the display-name with a valid string for the line. If rpi-type is present in the Remote-Party-ID header field, the proxy MUST verify the endpoint is entitled to the special status indicated; if not the proxy MUST remove the rpi-type token from the header. If the proxy does not recognize the rpi-type token value, it MUST remove that token from the header.

3.3.2 DCS-TRACE-PARTY-ID

In the telephone network, calling identity information is also used to support regulatory requirements such as the Customer Originated Trace service, which provide the called party with the ability to report obscene or harassing phone calls to law enforcement. This service is provided independent of caller-id, and operates even if the caller requested anonymity. The calling party is here identified as the station originating the call. In order for this service to be dependable, the called party must be able to trust that the calling identity information being presented is valid.

To initiate a customer-originated-trace from an untrusted UAC, an additional header is defined for the INVITE request sent from the untrusted UAC to its proxy. This header is called Dcs-Trace-Party-ID, and does not appear in any other request or response. The proxy receiving a properly formed INVITE request with this header performs the service-provider-specific functions of recording and reporting the caller identity for law enforcement action. The proxy then completes the call to either an announcement server or

to the service-provider's business office to collect further information about the complaint. A trusted UAC does not use this header, as it initiates this action locally.

3.3.2.1 Syntax

The BNF description of this header is:

Dcs-Trace-Party-ID = "Dcs-Trace-Party-ID" ":" "<" addr-spec ">"

Addr-spec contains a URL that identifies the remote endpoint. Addr-spec typically contains a tel: URL giving the identity of the remote endpoint, or a DCS-URL with a private-param when privacy was requested by the remote endpoint. This URL SHOULD be the value received in the Dcs-Remote-Party-ID header of the harassing call.

3.3.2.2 Procedures at an Untrusted User Agent Client (UAC)

The UAC MUST insert a Dcs-Trace-Party-ID header into the initial INVITE message for a customer-originated-trace request. The UAC MUST use a Request-URI with username of "call-trace" and host identifying the provisioned proxy for the untrusted UA.

3.3.2.3 Procedures at a Trusted User Agent Client (UAC)

A trusted UAC performs the customer-originated-trace in a manner similar to the originating proxy, described below. A trusted UAC MUST NOT include this header in any request.

3.3.2.4 Procedures at an Untrusted User Agent Server (UAS)

This header MUST NOT appear in any response sent by a UAS.

3.3.2.5 Procedures at a Trusted User Agent Server (UAS)

This header MUST NOT appear in any request sent by a UAS.

3.3.2.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to an untrusted endpoint, performs both sets of procedures.

A proxy that is neither an originating proxy nor a terminating proxy MUST pass the Dcs-Remote-Party-ID header unchanged.

3.3.2.6.1 Procedures at Originating Proxy

If the Dcs-Trace-Party-ID header is present in the initial INVITE request from the UAC, and the Request-URI of the INVITE has username of "call-trace" and host of the originating proxy, the originating proxy MUST perform the service-provider-specific functions of recording and reporting the caller identity for law enforcement action. The proxy then MUST direct the call to either an announcement server or to the service-provider's business office to collect further information about the complaint.

The originating proxy MUST remove the Dcs-Trace-Party-ID header from the INVITE before sending the request to another proxies or UAS.

3.3.2.6.2 Procedures at Terminating Proxy

This header MUST NOT appear in any request or response sent by a proxy.

3.3.3 DCS-ANONYMITY

When a call is placed, the calling identity delivery services reveal privacy information to the called party, and the calling party therefore has the option to block the delivery of this information to the called party. In the PSTN, this is typically achieved by subscribing to a Calling Identity Delivery Blocking service but can be done on an individual call basis as well. When the Calling Identity Delivery Blocking Service is invoked, information about the calling party is still passed through the trusted intermediaries, however presentation restriction indicators are set in the signaling messages to signal the far-end side, that the calling identity information is not to be provided to the called party.

More generally, we may say that the service provided is that of preventing the called party from obtaining information about the calling party that may either be used to identify the party or reveal location information about the party. In an IP environment, IP addressing information may provide the other party with information to reach or identify the calling party. IP addressing information may reveal some level of location information, for instance if one has knowledge of which addresses are deployed where, or by revealing that a given caller is using a different IP-address or address block than usual.

When such a privacy service is to be provided in a SIP environment, it leads to two requirements. First, calling identity information present in SIP messages must not be delivered in an intelligible form to the called party, yet it must be possible to determine the identity of the call originator even in the case where the call is routed through one or more untrusted intermediaries. Secondly, when using SIP in an IP environment, IP addressing information must be able to be hidden from the other party. Furthermore, in an IP environment, these requirements apply equally well in the opposite direction, i.e. the calling party may wish to identify the called party and the called party may have privacy concerns as well.

The Dcs-Anonymity extension allows an originating or terminating client to indicate the degree of privacy that should be provided by the service provider. This header is further described in [20].

3.3.3.1 Syntax

The BNF description of this header is:

Dcs-Anonymity = "Dcs-Anonymity" ":" #privacy-tag

privacy-tag = "Full" | "URI" | "Name" | "IPAddr" | "Off"

The value "URI" requests the endpoint's phone number not be available to the destination. The value "Name" requests the endpoint's name not be provided. The value "IPAddr" requests IP privacy such that the endpoint is not given the remote endpoint's IP address. The value "Full" requests URI blocking, Name

blocking, IP address privacy, and call-return blocking. Any combination of these values may appear in this header.

The value “Off” indicates no privacy is requested, and is the only tag if present.

Absence of this header in a request or response is identical to a value “Off”.

To maintain privacy of the initiator, an INVITE request send on the end-end signaling path SHOULD NOT contain a Dcs-Anonymity header.

3.3.3.2 Procedures at an Untrusted User Agent Client (UAC)

The originating UAC MAY insert a Dcs-Anonymity header into the initial INVITE message for a new call.

The value “Off” indicates no privacy is requested, and MUST be the only tag if “Off” is present.

If the endpoint has not requested privacy, and the Dcs-Anonymity header is present, it MUST be Off.

If the endpoint has requested privacy, the UAC MUST insert a Dcs-Anonymity header, and it MUST be one or more of Full, URI, Name, or IPAddr.

An untrusted UAC receiving a response to an initial INVITE SHOULD use information in the Dcs-Remote-Party-ID header for called-identity delivery, which is the information verified by the proxies. See section 3.3.1.2.

3.3.3.3 Procedures at a Trusted User Agent Client (UAC)

The originating UAC MAY insert a Dcs-Anonymity header into the initial INVITE message for a new call.

The value “Off” indicates no privacy is requested, and MUST be the only tag if “Off” is present.

If the endpoint has not requested privacy, and the Dcs-Anonymity header is present, it MUST be Off.

If the endpoint has requested privacy, the UAC MUST insert a Dcs-Anonymity header, and it MUST be one or more of Full, URI, Name, or IPAddr.

A trusted UAC MUST use the value of the Dcs-Anonymity header of the response to an initial INVITE to limit any display of the Dcs-Remote-Party-ID information. If the Dcs-Anonymity header of the response to an initial INVITE includes URL or Full, the UAC MUST NOT reveal the value of addr-spec. If the Dcs-Anonymity header of the response to an initial INVITE includes Name or Full, the UAC MUST NOT reveal the value of display-name.

3.3.3.4 Procedures at an Untrusted User Agent Server (UAS)

The terminating UAS MAY insert a Dcs-Anonymity header into the first non-100 response to the initial INVITE message for a new call.

The value “Off” indicates no privacy is requested, and MUST be the only tag if “Off” is present.

If the endpoint has not requested privacy, and the Dcs-Anonymity header is present, it MUST be Off.

If the endpoint has requested privacy, the UAS MUST insert a Dcs-Anonymity header, and it MUST be one or more of Full, URI, Name, or IPAddr.

An untrusted UAS receiving an initial INVITE SHOULD use information in the Dcs-Remote-Party-ID header for calling-identity delivery, which is the information verified by the proxies. Information contained in the display-name string in the From header MUST NOT be used as authenticated calling-identity, as this is supplied by the originating user and not verified by the proxies. See section 3.3.1.4.

3.3.3.5 Procedures at a Trusted User Agent Server (UAS)

The terminating UAS MAY insert a Dcs-Anonymity header into the first non-100 response to the initial INVITE message for a new call.

The value “Off” indicates no privacy is requested, and MUST be the only tag if “Off” is present.

If the endpoint has not requested privacy, and the Dcs-Anonymity header is present, it MUST be Off.

If the endpoint has requested privacy, the UAS MUST insert a Dcs-Anonymity header, and it MUST be one or more of Full, URI, Name, or IPAddr.

A trusted UAS MUST use the value of the Dcs-Anonymity header in the initial INVITE to control the display of the Dcs-Remote-Party-ID information. If the Dcs-Anonymity header of the initial INVITE includes URL or Full, the UAS MUST NOT reveal the value of addr-spec. If the Dcs-Anonymity header of the initial INVITE includes Name or Full, the UAS MUST NOT reveal the value of display-name.

3.3.3.6 Procedures at Proxy

The proxy MUST delete this header before passing the INVITE message or response to the INVITE message to an untrusted client or proxy.

Procedures for updating the value of Dcs-Remote-Party-ID header based on the value of Dcs-Anonymity are given in sections 3.3.1.6.1 and 3.3.1.6.2.

3.3.4 DCS-MEDIA-AUTHORIZATION

Enhanced quality of service, as required for high-grade voice communication, needs special authorization for better than 'best-effort' service. Without such a capability, it is possible that a single berserk IP telephony device can cause denial of service to a significant number of others.

The Sip Proxy authorizes the Media data flow to/from an untrusted SIP Client and supplies to the Client a Media-Authorization-Token, which is to be used for authorization when bandwidth is requested for the data-stream.

When the Client is ready to send the media data-stream to the other end-point, it first requests bandwidth, using the Authorization-Token it received from its SIP-Proxy.

The Dcs-Media-Authorization extension conveys the token needed in resource reservation messages to identify the connection and to associate the resources with an authorized connection. This header is further described in [21].

3.3.4.1 Syntax

The BNF description of the Dcs-Media-Authorization header is as follows:

Media-Auth = “Dcs-Media-Authorization” “:” Media-auth-token
Media-Auth-token = 1*hex

The token is a general format value, used to request resources, whose format is dependent on the particular resource reservation scheme utilized by the UAC/UAS, and authorized by the proxy. For DCS, this token is a gate-identification, used in the resource reservation signaling as specified in D-QoS [4].

3.3.4.2 Procedures at an Untrusted Initiator

The initiator of a request that requires QoS authorization (the UAI) may be either the UAS or UAC of an active call, or the UAC for a new call.

The Media-Auth-Token, contained in the Media-Authorization header, is sent by the proxy to an untrusted UAI in the first non-100 response message to an INVITE requiring QoS authorization.

The UAI SHOULD use the Media-Auth-Token from the first non-100 response message when requesting bandwidth for the Media data streams, both during the initial reservation and for subsequent refreshes.

3.3.4.3 Procedures at a Trusted Initiator

The initiator of a request that requires QoS authorization (the UAI) may be either the UAS or UAC of an active call, or the UAC for a new call.

A UAI located within the trust boundary of the service provider interfaces directly with an entity that authorizes the QoS for the media streams.

3.3.4.4 Procedures at an Untrusted Recipient

The recipient of a request that requires QoS authorization (the UAR) may be either the UAS or UAC of an active call, or the UAS for a new call.

The Media-Auth-Token, contained in the Media-Authorization header, is sent by the proxy to an untrusted UAR in the INVITE request that requires QoS authorization.

The UAR SHOULD use the Media-Auth-Token from the initial INVITE request when requesting bandwidth for the Media data streams, both during the initial reservation and for subsequent refreshes.

3.3.4.5 Procedures at a Trusted Recipient

The recipient of a request that requires QoS authorization (the UAR) may be either the UAS or UAC of an active call, or the UAS for a new call.

A UAR located within the trust boundary of the service provider interfaces directly with an entity that authorizes the QoS for the media streams.

3.3.4.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an initiating proxy, and (2) the procedures at a receiving proxy.

The initiating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The receiving proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to a non-trusted endpoint, performs both sets of procedures.

A proxy that is neither an initiating proxy nor a receiving proxy has no function in media authorization.

3.3.4.6.1 Procedures at Initiating Proxy

The Initiating Proxy authenticates the UAI, and verifies the UAI is authorized to receive the requested level of QoS. In cooperation with an entity that authorizes QoS for the media streams, they generate a Media-Auth-Token that contains sufficient information for the originating client to get the authorized bandwidth for the media streams.

The Initiating Proxy MUST insert the Media-Authorization header in the first non-100 response message to the initial INVITE, or mid-call INVITE that require a QoS change, that it sends to UAI.

3.3.4.6.2 Procedures at Receiving Proxy

The Receiving Proxy authenticates the UAR, and verifies the UAR is authorized to receive the requested level of QoS. In cooperation with an entity that authorizes QoS for the media streams, they generate a Media-Auth-Token that contains sufficient information for the destination servier to get the authorized bandwidth for the media streams.

The Receiving Proxy MUST insert the Media-Authorization header in the initial INVITE message, or mid-call INVITE that requires a QoS change, that it sends to UAR.

3.3.5 DCS-GATE

The Dcs-Gate header extension is used only on requests and responses between proxies. It never is sent to, nor sent by, an untrusted UAC/UAS.

The proxy-proxy signaling establishes a synchronization path that may be required by the Dynamic Quality of Service (D-QoS) specification [4] to coordinate the release of resources of the call. As per the D-QoS specification, the CMTS monitors the packet flow, and generates a Gate-Close message in response to either an explicit close request from the MTA/RGW, or when an equipment or facility failure causes the connection to be broken. This Gate-Close message is directed either to the local CMS/Agent, or to the remote CMS/Agent, or to the CMTS serving the remote MTA, depending on the capabilities of the endpoints. When a CMS/Agent receives such a Gate-Close message, it considers it identical to a call termination request.

The Dcs-Gate header is used between proxies, and conveys the location of the remote gate, identity of the gate, and the security key to be used in gate coordination messages. This header is further described in [24].

3.3.5.1 Syntax

The BNF description of the Dcs-Gate header is as follows:

Dcs-Gate = "Dcs-Gate" ":" hostport “/” Gate-ID
[“;” Gate-Key “;” Gate-CipherSuite]
[Gate-strength-token]

Gate-ID = 1*alphanum

Gate-Key = 1*alphanum

Gate-CipherSuite = token

Gate-strength-token = “required” | “optional”

Hostport gives the IP address or FQDN of the CMTS/EdgeRouter that enforces the QoS, or the endpoint system that simulates the gate coordination exchange on behalf of an edge router.

Gate-ID is a token used at the system named in the Hostport parameter to identify the particular session. For DCS systems, it is a 32-bit quantity encoded as an 8-character string of digits 0-9 and letters a-f.

Gate-Key is a character string that provides keying information to the system named in the hostport parameter. The method of deriving the actual keys for the gate coordination messages, and the security procedures, are described in [2].

Gate-CipherSuite is a character string that gives the type of encryption algorithm that will be used to secure the gate coordination messages. See [2] for further definitition.

Gate-Strength-Token specifies whether the gate coordination is required or optional for the current session. Its use is described in the following sections.

3.3.5.2 Procedures at an Untrusted User Agent Client (UAC)

This header is never sent to an untrusted UAC, and is never sent by an untrusted UAC.

3.3.5.3 Procedures at a Trusted User Agent Client (UAC)

A UAC located within the trust boundary of the service provider performs the functions given in section 3.3.5.6.1.

3.3.5.4 Procedures at an Untrusted User Agent Server (UAS)

This header is never sent to an untrusted UAS, and is never sent by an untrusted UAS.

3.3.5.5 Procedures at a Trusted User Agent Server (UAS)

A UAS located within the trust boundary of the service provider performs the functions given in section 3.3.5.6.2.

3.3.5.6 Procedures at Proxy

The Dcs-Gate header MUST NOT appear in any message other than the initial INVITE request, or in the first non-100 response to that request. The proxy MUST remove the Dcs-Gate header in any request or response sent to an untrusted endpoint.

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to a non-trusted endpoint, does not generate a Dcs-Gate header.

A proxy that is neither an originating proxy nor a terminating proxy has no function in coordinating the commitment of resources.

3.3.5.6.1 Procedures at Originating Proxy

The originating proxy MUST insert a Dcs-Gate header in the initial INVITE message for a new call.

The originating proxy MUST identify the system that will perform gate coordination (either the proxy itself, or the CMTS controlling the media flow to the endpoint), and the identification token used at that system to identify the call. It MUST insert the IP address or FQDN of that system in the hostport parameter of the Dcs-Gate header, and the identification token as the Gate-ID.

The originating proxy MUST pick a security key and cipher suite for the gate coordination message exchange, and insert these values in the Dcs-Gate header.

If the system that will perform gate coordination is a CMTS, the strength token MUST be given as required. If the system that will perform gate coordination is the proxy itself, the strength token MAY be given as optional, or omitted.

3.3.5.6.2 Procedures at Terminating Proxy

The terminating proxy MUST identify the system that will perform gate coordination (either the proxy itself, or the CMTS controlling the media flow to the endpoint), and the identification token used at that system to identify the call. Gate coordination will be required for this call if (1) the strength token in the Dcs-Gate header in the initial INVITE indicates 'required', or (2) the system that will perform gate coordination at the destination is a CMTS.

If gate coordination is required for this call, the terminating proxy MUST include a Dcs-Gate header in the first non-100 response to the initial INVITE request. It MUST insert the IP address or FQDN of the system that will perform gate coordination in the hostport parameter of the Dcs-Gate header, and the identification token as the Gate-ID.

If gate coordination is not required for this call, the terminating proxy SHOULD NOT include a Dcs-Gate header in the first non-100 response to the initial INVITE request.

3.3.6 DCS-STATE

The Distributed call signaling (DCS) architecture provides signaling support for creating a session using a signaling scheme so that call state is distributed to the clients.

There are three kinds of state associated with a call - transaction state, connection state, and call state. The goal with managing state is to store state about the call at places where it is needed.

Transaction state includes information about the current request and how it is being processed, how the response needs to be routed, and any partial processing done with the request that is needed in forming the response. SIP presently defines a mechanism by which transaction state associated with a request can be passed to the endpoint and returned in the response to the proxy – through the use of via header encryption. A proxy that encrypts the via headers can include other transaction state in the encrypted string, which can be decrypted and recovered in every provisional and final response generated to this request. DCS uses this mechanism to provide anonymity to the caller.

Connection state refers to the state associated with the media path. This includes the characteristics of the flow, admission control and policing parameters and is stored in devices in the network/media path where admission control and policing decisions are made. Connection state also includes billing information, and the unique call-identifying token (also known as the billing-correlation-id) used by the billing subsystem to correlate event records generated by the call. Connection state is distributed to the network elements during the call setup phase, and not stored by the proxy during the call.

Call state refers to endpoint identification, caller and callee preferences that affect active call characteristics, and network and transactions state hooks or identifiers in the active call that can be used by the proxy to modify the characteristics of the call. By using this mechanism, the proxies can offer the full range of required services, yet remain stateless during the call.

This state information is distributed to the endpoints during call setup through the use of Dcs-State headers. The state information may also be encrypted, signed, and contain an integrity check value, to guarantee detection of tampering by the untrusted client/server.

If the client wishes to change call characteristics that affect bearer path and/or require billing changes, it passes the saved proxy encrypted and signed state information in a SIP INVITE request to its proxy server, which may verify integrity of the state and decrypt it. The proxy is then able to perform the requested action, just as if the proxy had maintained the call state information itself.

RFC2543[11] section 12.4.3 notes that even a stateless proxy caches the results of address translations to speed forwarding of retransmissions. When that cached information is placed in a Dcs-State header, it provides a mechanism analogous to the Record-Route/Route header mechanism for storing the routing of the signaling path between the call originator and destination.

3.3.6.1 Syntax

The Dcs-State header is described by the following BNF:

Dcs-State = "Dcs-State" ":" 1#(host “;” State-Token *(“;” State-Token))

State-Token = token [“=” (*token | quoted-string)]

The host field identifies the proxy that inserted the state information, and may be either a IPv4Address or a FQDN. If the proxy generating the Dcs-State header is part of a multiple CPU cluster with multiple IP address, it SHOULD use the FQDN instead of IPv4Address so that all systems in the cluster will recognize their Dcs-State header.

A request or response MAY contain multiple Dcs-State headers.

The set of state tokens saved by a proxy is a local matter to the proxy, and is not part of the interface between proxies. Therefore, the particular token names, and the format of the quoted strings or token sequences assigned to each, is not specified. An example of a Dcs-State header is:

> Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0

A downstream proxy may combine Dcs-State headers generated by multiple proxies into a single Dcs-State header. This may be done simply by including an additional State-Token such as ‘upstream-state=”previous-state-header-value”’. The combined State-Token is returned from the client to the proxy, where it is restored for the upstream proxies.

State-tokens are typically encrypted and signed before delivering them to an untrusted endpoint, and the last token in a sequence is likely to be an integrity check over the previous. Such a signed&sealed Dcs-State header is typically a single State-Token, such as ‘state=”ascii-encoding-of-encrypted-State-Tokens”’. The quoted string here contains an encoding of an encrypted structure containing multiple separate State-Tokens needed by the proxy to perform the mid-call features. The encrypted structure is returned from the client to the Proxy, where it is decrypted, checked for tampering, and restored for use. An example of an encrypted combined state token is:

> Dcs-State: Host(dp-t.provider); state=”{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state=”Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0”}$_K$”

3.3.6.2 Procedures at User Agent Clients (UACs) and User Agent Servers (UASs)

Every UAC and UAS, whether within the trust boundary or outside the trust boundary, MUST save the Dcs-State headers received in the initial INVITE and first non-100 response for the duration of the call, along with the call-leg identification (From, To, and Call-ID values).

On subsequent request messages (e.g. INVITEs and ACKs) or responses (e.g. 180-Ringing, 200-OK) sent to the proxy, a Dcs-State header is included if the call-leg identification matches those associated with the saved Dcs-State header.

On subsequent INVITE request messages that include a Dcs-Also header, where the Dcs-Also header has attached a Call-ID header and a Dcs-Replaces header (i.e. Dcs-Also: URL ? Call-ID=XX & Dcs-Replaces=YY), a Dcs-State header MUST be attached to the Dcs-Also if (1) the Call-ID in the Dcs-Also header matches the Call-ID of the Dcs-State, and (2) the Dcs-Replaces in the Dcs-Also header matches either the From or To of the Dcs-State.

When a call-leg identified by a From, To, and Call-ID ends, the client MAY delete saved Dcs-State headers associated with this call-leg.

3.3.6.3 Procedures at Proxy

Proxy procedures for handling Dcs-State are based on a few simple rules, given below. Application of these rules to achieve the necessary functionality is not so straightforward, and is described in the next section.

A Proxy MAY generate one or more Dcs-State headers, and include it (or them) in any request or response.

A Dcs-State header with hostname of another proxy MUST be passed on.

A Dcs-State header with hostname matching the proxy MAY be discarded.

A Proxy MAY encrypt the Dcs-State headers, in a manner analagous to that of Via header encryption described in SIP section 13.1.5. As only the proxy that encrypts the field will decrypt it, the algorithm chosen is entirely up to the proxy implementor. Two methods satisfy these requirements:

- The proxy keeps a cache of Dcs-State header(s), and replaces the Dcs-State header(s) with a single Dcs-State header identified with the caching proxy that contains an index into the cache. On the reverse path, take the Dcs-State header(s) from the cache rather than the message.
- The proxy MAY use a secret key to encrypt the Dcs-State header(s), and an appropriate checksum in any such message with the same secret key. The checksum is needed to detect whether successful decoding has occurred. The encrypted Dcs-State header(s) are included in the request/response as a single Dcs-State header identified with the encrypting proxy. This is the preferred solution.

3.3.6.4 Application of this header in DCS

Three sets of proxy procedures are defined: (1) the procedures at an originating proxy, (2) the procedures at a terminating proxy, and (3) the procedures at a tandem proxy. For each, subsections define the procedures in handling a request and in handling a response.

The originating proxy is a proxy that received the INVITE request from an untrusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to an untrusted endpoint.

A Tandem proxy is one that is neither an originating proxy nor a terminating proxy.

3.3.6.4.1 Procedures for handling a Request at an Originating Proxy

The originating proxy SHOULD add a Dcs-State header to the initial INVITE request, including state information it needs to process mid-call signaling messages that originate at the called party. In order to achieve stateless operation, the Proxy SHOULD include in this Dcs-State header the Gate location and identifier (for retrieving the billing identifier and accounting information), the value of the Request-URI needed to identify the call originator for mid-call messages, and the original destination and redirection count (in support of Electronic Surveillance).

3.3.6.4.2 Procedures for handling a Request at a Tandem Proxy

When a tandem proxy is specifically addressed in a Request-URI, it SHOULD add a Dcs-State header to the initial INVITE, including information it needs to route mid-call signaling messages that originate at the called party. This information SHOULD include the translation information to enable the tandem proxy to send the message to the originating proxy.

When a tandem proxy is used to assist the originating and terminating proxies to exchange request and response messages, due to reasons such as security associations, and the Request-URI does not specifically address the tandem proxy, the tandem proxy SHOULD NOT add a Dcs-State header.

3.3.6.4.3 Procedures for handling a Request at a Terminating Proxy

The terminating proxy MUST add a Dcs-State header to the initial INVITE request, including information it needs to process mid-call signaling messages that originate at the called party. The terminating proxy MUST sign and encrypt the data with its private key, and append the header to the INVITE that is formed for the destination endpoint. This state information SHOULD contain gate location and gate-id at $CMTS_T$, the value of the Request-URI needed to identify the path back to the call originator, the Dcs-Laes and Dcs-Redirect header values, if present, and a concatenation of all other Dcs-State headers from other proxies in the path. Additional information such as Billing-ID, and Billing-Info, MAY be contained in the Dcs-State string, or MAY be obtained when needed from the gate parameters.

3.3.6.4.4 Procedures for handling a Reponse at a Terminating Proxy

The terminating proxy SHOULD add a Dcs-State header to the first non-100 response to an initial INVITE, including state information it needs to process mid-call signaling messages that originate at the calling party. In order to achieve stateless operation, the terminating proxy SHOULD include in this Dcs-State header the Gate location and identifier (for retrieving the billing identifier and accounting information), the Request-URI after translation was completed, and the original remote-party and redirection count (in support of Electronic Surveillance).

3.3.6.4.5 Procedures for handling a Response at a Tandem Proxy

When a tandem proxy is specifically addressed in a Request-URI, it SHOULD add a Dcs-State header to the first non-100 response to an initial INVITE, including information it needs to route mid-call signaling messages that originate at the calling party. This information SHOULD include the Request-URI after translation was completed, needed to reach the same destination endpoint for further requests.

When a tandem proxy is used to assist the originating and terminating proxies to exchange request and response messages, due to reasons such as security associations, and the Request-URI does not specifically address the tandem proxy, the tandem proxy SHOULD NOT add a Dcs-State header.

3.3.6.4.6 Procedures for handling a Response at an Originating Proxy

The originating proxy MUST add a Dcs-State header to the first non-100 response to an initial INVITE, including information it needs to process mid-call signaling messages that originate at the calling party. This information SHOULD include the gate location and Gate-ID at $CMTS_O$, needed to support mid-call codec changes, and SHOULD include the Request-URI after translation was completed, needed to reach the same destination endpoint for further requests. The originating proxy MUST then take the set of all Dcs-State headers and form a single Dcs-State header containing them all. Using its private key, the originating proxy MUST sign and encrypt the resulting Dcs-State header, and MUST replace all Dcs-State headers with this single Dcs-State header in the response formed for the UAC.

3.3.7 DCS-ALSO and DCS-REPLACES

This section describes a set of extensions to SIP which allow for various call control services. Example services include blind transfer, transfer with consultation, and ad-hoc conferencing. For the various services described here, we overview the requirements for the service, and specify the protocol functions needed to support it. We then define a basic set of SIP primitives that can be used to construct these services, and others.

This section describes extensions to SIP for providing call control services. Call control services relate to participant management. These services are all built on the basic blocks of adding and removing users from a call. Examples include transfer (the simultaneous removal and addition of a member from a call), multi-party calling, call bridging, and ad-hoc bridged conferences. Our aim is to provide a general set of tools, which can be used to construct, at a minimum, a core set of services, but be potentially useful as building blocks for future services. To accomplish this goal, we begin by overviewing the requirements for each of the core services, then outline the basic primitives that we have concluded are needed. The following subsection formally defines these primitives through new headers and UA behavior.

In the blind transfer service, two parties are in an existing call. One party (the transferring party) wishes to terminate the call with the other party (the transferred party), and at the same time transfer them to another party (the transferred-to party).

Transfer with consultation is similar to blind transfer. However, the transferring party first contacts the transferred-to party to approve the transfer through multimedia communication. Pending approval, the transferring party then simultaneosly disconnects from the transferred-to and transferred parties, and connects the transferred and transferred-to parties. The transferring and transferred parties stay connected if, for some reason, the transfer fails.

In three-way-calling service, a user A has a call in progress with B, and a separate call in progress with C. These calls are unrelated, with different Call-ID's. From this double call scenario, the conference out of consultation service allows the calls to be merged, resulting in a single, bridged conference.

Two additional headers are defined to provide these services, Dcs-Also and Dcs-Replaces. Various combinations of these with other existing headers are used to provide the call control services listed above, and can be used to provide a number of other services beyond those listed.

The Dcs-Also extension advises the recipient to issue INVITE requests to the URLs listed. The Dcs-Replaces extension advises the recipient to issue a BYE request to an existing call leg. When combined, the Dcs-Also function is performed prior to the Dcs-Replaces function.

Consider a call from party A to party B. Party B wishes to blind transfer the call to party C. B sends an INVITE A, Also: C, Replaces: B. Party A calls C, then issues a BYE to B, completing the transfer.

Transfer with consultation proceeds similarly. Party B establishes a second connection to party C, for consultation. When ready to transfer, B sends an INVITE A, Also: C, Replaces: B. Additional headers in the Also: C are used to identify the particular call leg between B and C that is to be redirected.

Three-way-calling, or ad-hoc conferencing, involves a bridge service in DCS. Two prior calls exist, from party A to party B, and from party A to party C; party A wishes to join them in an ad-hoc conference. A sends an INVITE bridge-service, Also: B, C. Additional headers in the Also: B identify the particular call leg between A and B that is to be replaced, and additional headers in the Also: C identify the particular call leg between A and C.

Throughout the following descriptions, INVITE(Also) refers to an INVITE message that contains a Dcs-Also header; INVITE(Replace) refers to an INVITE message that contains a Dcs-Replaces header, and INVITE(Also,Replace) refers to an INVITE message that contains both.

3.3.7.1 Syntax

Dcs-Also = "Dcs-Also" ":" DCS-URL *[“,” DCS-URL]

Dcs-Replaces = “Dcs-Replaces” “:” SIP-URL *[“,” SIP-URL]

An example of a simple Dcs-Also header, as used in the blind transfer service, is:

INVITE sip:555-1111@dp.provider
Dcs-Also: sip:555-3333@dp.provider
Dcs-Replaces: sip:555-2222@dp.provider

This alters an existing call from 555-1111 to 555-2222, transferring the original caller (555-1111) to the new destination 555-3333.

An example of a more complex Dcs-Also header, as used in the three-way-calling service, is:

INVITE sip:bridge@dp.provider
Dcs-Also: sip:555-2222@dp-b.provider ?
Call-ID=call-from-555-1111-to-555-2222 &
Dcs-Replaces=sip:555-1111@dp-o.provider

This causes the bridge service to initiate a call to 555-2222, with the new call having the same Call-ID as the original one from 555-1111, and replacing 555-1111.

3.3.7.2 Procedures at initiator

Dcs-Also and Dcs-Replaces provide tools by which many call control services may be built. For purposes of this specification, only three are specified at the initiator: blind transfer, consultative transfer, and ad-hoc conferencing. The procedures necessary to support these are specified at the recipient.

A mid-call change is sent by the initiator (UAI) which may be either the UAS or UAC for the active call. The destination of the mid-call change is the recipient (UAR).

3.3.7.2.1 Blind Transfer

To initiate a blind transfer, there MUST be an existing call from UAI to UAR. The identity of the desired transfer destination is given here as URL-T. The desired end result is a call from UAR to T. This is shown in Figure 4. From the starting state (left diagram) UAI sends an INVITE(also,replace) to UAR, who initiates an INVITE to T (middle diagram), then UAR terminates the call with UAI (rightmost diagram).

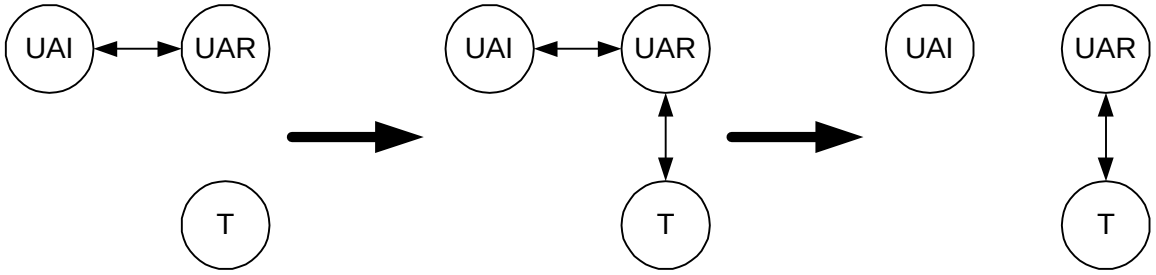


Figure 4: Call Transfer (Blind)

To initiate a blind transfer, the UAI MUST send an INVITE to UAR, with call-leg identification matching the existing call between UAI and UAR, that contains both a Dcs-Also header and a Dcs-Replaces header. The INVITE request MUST NOT contain an SDP.

The Dcs-Also header MUST contain URL-T. The Dcs-URL MAY contain headers for Dcs-Billing-ID, Dcs-Billing-Info, and Dcs-State. For this service, additional header parameters Call-ID and Dcs-Replaces MUST NOT be attached to the Dcs-Also header.

An originating UAC within the trust boundary of the service provider MUST include billing arrangements for the new call(s) to be made by the recipient, as a result of this message. The UAC MUST include Dcs-Billing-ID and Dcs-Billing-Info additional headers appended to the Dcs-Also header to provide this information. See section 3.3.9 for information about the billing headers. The originating UAC MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber. If found, the UAC MUST include a Dcs-LAES header in the INVITE. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MAY include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content, and MUST include a random string for use as a security key between the Delivery Functions.

Other additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The Dcs-Replaces header MUST contain a copy of the From or To header (whichever refers to UAI) from the existing UAI-UAR call: copy of From header if the UAI is the UAC and copy of To if the UAI is the UAS. Additional header parameters MUST NOT be attached to the Dcs-Replaces header.

The response to the INVITE is a non-200 value if the UAR was unwilling or unable to execute the request.

A 200-OK response to the INVITE indicates a successful initiation of the blind transfer; any error response leaves the original call intact. A BYE message from UAR indicates successful completion of the blind transfer.

3.3.7.2.2 Consultative Transfer

To initiate a consultative transfer, there MUST be a call active from UAI to UAR, and a call active from UAI to the transfer destination, called here UAT. The Call-ID for the existing call from UAI to UAR is denoted UAI- UAR. This is shown in Figure 5. From the starting state (left diagram) UAI sends an INVITE(also,replace) to UAT, who initiates an INVITE(replace) to UAR (next diagram), UAR terminates the call with UAI (next diagram), and UAT terminates the call with UAI (final diagram).

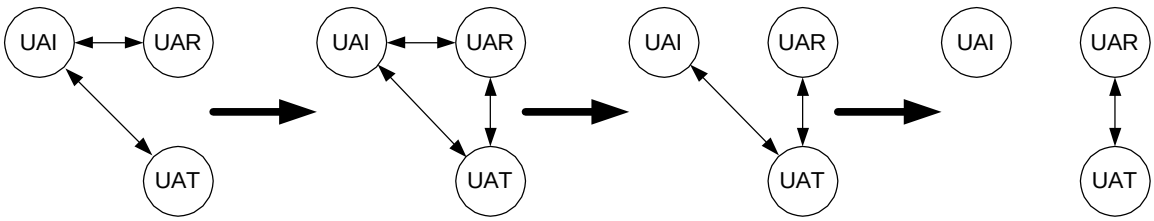


Figure 5: Call Transfer (Consultative)

To initiate a consultative transfer, the UAI MUST send an INVITE to UAT, with call-leg identification matching the existing call between UAI and UAT, that contains both a Dcs-Also header and a Dcs-Replaces header. The INVITE request MUST NOT contain an SDP.

The Dcs-Also header MUST contain the URL of UAR, and MUST have attached a Call-ID header with value UAI-UAR, and a Dcs-Replaces header giving the From or To header (whichever refers to UAI) for the existing call from UAI to UAR.

An untrusted UAC MUST attach the collection of Dcs-State headers for the call from UAI to UAR.

An originating UAC within the trust boundary of the service provider MUST include billing arrangements for the new call(s) to be made by the recipient, as a result of this message. The UAC MUST include Dcs-Billing-ID and Dcs-Billing-Info additional headers appended to the Dcs-Also header to provide this information. See section 3.3.9 for information about the billing headers. The originating UAC MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber. If found, the UAC MUST include a Dcs-LAES header in the INVITE. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MAY include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content, and MUST include a random string for use as a security key between the Delivery Functions.

Additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The Dcs-Replaces header MUST contain the value of the From or To header (whichever refers to UAI) from the existing call between UAI and UAT. Additional header parameter MUST NOT be attached to the Dcs-Replaces header.

The response to the INVITE is a non-200 value if UAT was unwilling or unable to execute the request.

A 200-OK response to the INVITE indicates a successful initiation of the transfer; any error response leaves the original call intact. BYE messages from UAR and from UAT indicate successful completion of the consultative transfer.

3.3.7.2.3 Ad-hoc Conference

To initiate an ad-hoc conference, there MUST be a two (or more) calls active from UAI, destinations denoted here by A and B. The Call-ID for the existing calls are denoted UAI-A and UAI-B. From the starting conditions (left diagram) UAI sends an INVITE(also) to a bridge (middle diagram), who sends INVITE(replace)s to A and B, A and B terminate their call with UAI (right diagram).

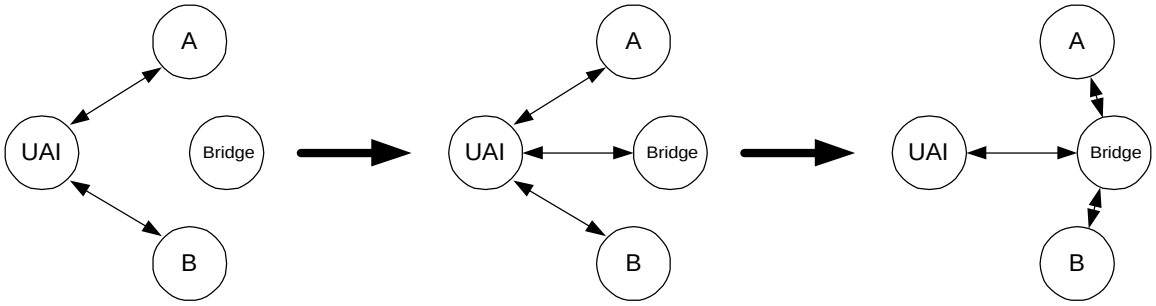


Figure 6: Ad-hoc Conferencing

To initiate an ad-hoc conference, the UAI MUST send an INVITE to a bridge server that contains at least two Dcs-Also headers. Each Dcs-Also header MUST contains the URL of one of the existing endpoints. This INVITE MUST contain an SDP message body, describing the media flow between the bridge and UAI.

The first Dcs-Also header MUST contain the URL of URL-A, and MUST have attached a Call-ID header with value UAI-A, and a Dcs-Replaces header giving the From or To header (whichever refers to UAI) for the existing call from UAI to A.

An untrusted UAC MUST attach the collection of Dcs-State headers for the call from UAI to A.

An originating UAC within the trust boundary of the service provider MUST include billing arrangements for the new call(s) to be made by the recipient, as a result of this message. The UAC MUST include Dcs-Billing-ID and Dcs-Billing-Info additional headers appended to the Dcs-Also header to provide this information. See section 3.3.9 for information about the billing headers. The originating UAC MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber. If found, the UAC MUST include a Dcs-LAES header in the INVITE. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MAY include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content, and MUST include a random string for use as a security key between the Delivery Functions.

Additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The second Dcs-Also header MUST contain similar information for the call from UAI to B.

Call establishment procedes normally for the new call between UAI and bridge, including resource pre-conditions, ringing, etc. At the point when the normal call sends a 200-OK response, the bridge processes the Dcs-Also headers.

The response to the INVITE is a non-200 value if the bridge failed to establish a call leg with UAI, or if the bridge was unwilling or unable to execute the request.

A 200-OK response to the INVITE indicates a successful initiation of an ad-hoc conference. Receipt of BYEs for the existing calls from UAI to A and UAI to B indicates a successful completion of the ad-hoc conference setup. An error response to the INVITE indicates a failure to establish the bridge. A 200-OK response to the INVITE, with one or more (but not all) BYE messages, indicates a failure of one or more parties to be transferred to the bridge; those calls are still active from UAI.

3.3.7.3 Additional Procedures at a Trusted Initiator

If both Dcs-Also and Dcs-Replaces headers are present, the UAI MUST verify that the initiating endpoint is authorized to perform this function. This typically means the initiator subscribed to either Call Transfer or Three-way-calling service.

The UAI MUST include billing arrangements for the new call(s) to be made by the recipient, as a result of this message. The UAI MUST include Dcs-Billing-ID and Dcs-Billing-Info additional headers appended to the Dcs-Also header to provide this information. See section 3.3.9 for information about the billing headers. The billing information for the first portion of the new call is the same as the existing call; the UAI MUST add a Dcs-Billing-Info header that provides billing information for the portion of the new call from the initiator to the specified destination, with the account number of the initiator.

The UAI MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber. If found, the UAI MUST include a Dcs-LAES header in the INVITE. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MAY include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content, and MUST include a random string for use as a security key between the Delivery Functions.

3.3.7.4 Procedures at an Untrusted Recipient

When a UAI sends an INVITE to the UAR containing a Dcs-Also header, it implies that the UAI wishes the UAR, in addition to the normal processing of the INVITE request, to send an INVITE to those parties listed in the Dcs-Also headers. If the INVITE message contains a Dcs-Replaces: header, it implies that the UAI wishes the UAR, following the normal processing of the INVITE request and Dcs-Also headers (if present), to send a BYE message for the call leg identified by the Dcs-Replaces header value.

The UAR, upon receiving an INVITE request, MUST first establish the new call, if the call-leg identification does not match an existing call at the UAR. This phase completes when the UAR would normally send the 200-OK to the INVITE request.

Second, the UAR MUST check the Dcs-Also headers for any condition that would prevent the initiation of the sessions requested (such as syntax errors). The UAR MUST check the Dcs-Replaces headers to verify the existance of an active call-leg with a matching Call-ID (either a Call-ID attached to the Dcs-Replaces header, or the Call-ID of the INVITE message itself) and a participant (either the From or To) matching the SIP-URL in the Dcs-Replaces header. A UAR that is not capable of performing local bridging of media streams SHOULD reject an INVITE that would result in two or more call legs with the same Call-ID upon completion. If these checks are successful, UAR responds to the INVITE with a 200-OK final response.

Third, the UAR MUST process any Dcs-Also headers present in the INVITE request. To do so, the UAR initiates an INVITE request with the Request-URI the value of the DCS-URL in the Dcs-Also header. Any additional headers attached to the Dcs-Also header MUST override the normal headers that would be included in this request. For example, a Call-ID header attached to the Dcs-Also means the UAR MUST use that value, and only that value, for the Call-ID header in the initiated INVITE. Headers that would not normally be included in the INVITE request, but which appear attached to the Dcs-Also, MUST be included in the INVITE request. Where multiple Dcs-Also headers are present, each MUST generate a separate INVITE request. This phase completes on receipt of a 200-OK from all of the initiated INVITEs.

Finally, the UAR MUST process any Dcs-Replaces headers present in the INVITE request. UAR sends a BYE request to the matching active call-leg identified earlier. This phase completes on receipt of a 200-OK from all the initiated BYEs.

3.3.7.5 Procedures at a Trusted Recipient

Procedures at a trusted recipient are identical to those described in 3.3.7.4.

3.3.7.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to a non-trusted endpoint, performs both sets of procedures.

A proxy that is neither an originating proxy nor a terminating proxy has no function in manipulating existing calls.

3.3.7.6.1 Procedures at Originating Proxy

If both Dcs-Also and Dcs-Replaces headers are present, the originating proxy MUST verify that the initiating endpoint has is authorized to perform this function. This typically means the initiator subscribed to either Call Transfer or Three-way-calling service.

The url-parameter "private" MAY appear in the Dcs-Also headers. When identified by the hostname of the URL, the proxy MUST decode/decrypt the username, and replace the URL with a new DCS-URL. If the replacement DCS-URL contains a private-param, it MUST identify a different hostname than the current proxy. If the proxy is unable to decode/decrypt the username, it MUST reject the call attempt with an appropriate 4xx error code.

The originating proxy MUST include billing arrangements for the new call(s) to be made by the recipient, as a result of this message. The proxy MUST include Dcs-Billing-ID and Dcs-Billing-Info additional headers appended to the Dcs-Also header to provide this information. See section 3.3.9 for information about the billing headers. If a Dcs-State header is attached to the Dcs-Also, it MUST be used in calculating the billing information for the new calls. Otherwise, the Dcs-State header in the INVITE message MUST be used for calculating billing information. The Dcs-State header provides the billing information for the first portion of the new call; the originating proxy MUST add a Dcs-Billing-Info header that provides billing information for the portion of the new call from the initiator to the specified destination, with the account number of the initiator.

The originating proxy MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber. If found, the proxy MUST include a Dcs-LAES header in the INVITE. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MAY include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content, and MUST include a random string for use as a security key between the Delivery Functions.

All Dcs-State headers MUST be removed from the Dcs-Also header.

3.3.7.6.2 Procedures at Terminating Proxy

The terminating proxy MUST form a private-param for the recipient endpoint, and include this private-param in the Dcs-Also header in the INVITE message passed to the endpoint. This private-param

SHOULD contain the following information: 1) the destination URL, 2) the value of Dcs-Billing-ID, 3) the sequence of Dcs-Billing-Info values, which indicate the complex charging arrangement for the new call, 4) an expiration time very shortly in the future, to limit the ability of the recipient to re-use this private-param for multiple calls, and 5) the electronic surveillance information, if present.

Any remaining headers MUST be carried in the Dcs-Also header, using the SIP syntax for optional headers attached to a SIP-URL.

3.3.8 DCS-OSPS

Some calls have special call processing requirements that may not be satisfied by normal user agent call processing. For example, when a user is engaged in a call and another call arrives, such a call might be rejected with a busy indication. However, some PSTN operator services require special call processing. In particular, the Busy line verification (BLV) and Emergency interrupt (EI) services initiated by an operator from an Operator Services Position System (OSPS) on the PSTN network have such a need.

In order to inform the SIP user agent that special treatment should be given to a call, we use a new OSPS header field, which may be set to a value indicating when a special type of call processing is requested. We define two values in this header, namely "BLV" for busy line verification and "EI" for emergency interrupt.

If the user agent decides to honor such a request, the response of the user agent to an INVITE with either "BLV" or "EI" will not be a busy indication. When such a request is received, the user agent may look at the Remote-Party-ID, and decide only to honor the request if "rpi-type" is "operator" and Remote-Party-ID was authenticated by the user agent's proxy.

3.3.8.1 Syntax

Dcs-OSPS = “Dcs-OSPS” “:” OSPS-Tag

OSPS-Tag = “BLV” | “EI” | token

The OSPS-Tag value of “token” is defined for extensibility, and is reserved for future use.

3.3.8.2 Procedures at an Untrusted User Agent Client (UAC)

The Dcs-OSPS header MUST NOT be sent in a request from an untrusted UAC.

3.3.8.3 Procedures at a Trusted User Agent Client (UAC)

This header is typically only inserted by a Media-Gateway-Controller that is controlling a Media Gateway with special MF trunk connections to a PSTN OSPS system. This trunk group is usually referred to as a BLV-trunk group, and employs special signaling procedures that prevent inadvertant use. Calls originating at the PSTN OSPS system are sent over this trunk group, and result in an INVITE request with the OSPS header.

This header MAY be sent in an INVITE request, and MUST NOT appear in any message other than an INVITE request.

OSPS-Tag value “BLV” MUST NOT appear in any INVITE other than an initial INVITE request establishing a new session.

OSPS-Tag value “EI” MUST NOT appear in any INVITE request other than a subsequent INVITE within a pre-existing session established with the OSPS-Tag value of “BLV”.

3.3.8.4 Procedures at an Untrusted User Agent Server (UAS)

If the UAS receives an INVITE request with an OSPS-Tag, call-leg identification that matches an existing call, and the existing call was not established with the OSPS-Tag, it MUST reject the request with a 409-Conflict error code. If the UAS receives an INVITE request with an OSPS-Tag value of “EI”, with call-leg identification that does not match an existing call, it MUST reject the request with a 409-Conflicct error code.

If the UAS receives an INVITE that contains an OSPS-Tag value of “BLV” and is not willing to cooperate in offering this service, it MUST reject the request with a 403-Forbidden error code. Otherwise, the UAS MUST verify the Dcs-Remote-Party-ID header contains a rpi-type token with value “operator.” If the call is not from a service-provider-certified operator, it SHOULD be rejected with a 401-Unauthorized error code.

The UAS SHOULD NOT reject an INVITE with a BLV OSPS-Tag due to a busy condition. The UAS MUST NOT respond with a 3xx-Redirect error code to an INVITE with a BLV OSPS-Tag. The UAS SHOULD NOT alert the user of the incomming call attempt if the BLV OSPS-Tag is present in the INVITE.

If an INVITE with OSPS-Tag of “BLV” is accepted (meeting all QoS pre-conditions, etc.), the UAS MUST send an audio stream on this conection to the address and port given in the SDP of the INVITE. The UAS MAY perform a mixing operation between the two ends of an active call. The UAS MAY send a copy of the local voice stream, and (if no activity on the local voice stream) send a copy of the received voice stream. If the state of the UAS is idle, the UAS SHOULD send a stream of silence packets to OSPS. If the state of the UAS is ringing or ringback, the UAS SHOULD send a ringback stream to OSPS.

If an INVITE with OSPS-Tag of “EI” is accepted, the UAS MUST enable communication between the UAC and the local user. The UAS MAY put any existing call on hold, or initiate an ad-hoc conference.

3.3.8.5 Procedures at a Trusted User Agent Server (UAS)

The procedures at a trusted UAS are identical to those described in 3.3.8.4.

3.3.8.6 Procedures at Proxy

There is no special processing of this header at proxies.

3.3.9 DCS-BILLING-ID and DCS-BILLING-INFO

In order to deploy a residential telephone service at very large scale across different domains, it is necessary for trusted elements owned by different service providers to exchange trusted information that conveys billing information and expectations about the parties involved in the call.

There are many billing models used in deriving revenue from telephony services today. Charging for telephony services is tightly coupled to the use of network resources. It is outside the scope of this document to discuss the details of these numerous and varying methods.

A key motivating principle of the DCS architecture is the need for network service providers to be able to control and monitor network resources; revenue may be derived from the usage of these resources as well as from the delivery of enhanced services such as telephony. Furthermore, the DCS architecture recognizes the need for coordination between call signaling and resource management. This coordination ensures that users are authenticated and authorized before receiving access to network resources and billable enhanced services.

Proxies have access to subscriber information and act as policy decision points and trusted intermediaries along the call signaling path. Edge routers provide the policy enforcement mechanism and also capture and report usage information. Edge routers need to be given billing information that can be logged with Record Keeping or Billing servers. The proxy, as a central point of coordination between call signaling and resource management, can provide this information based on the authenticated identity of the calling and called parties. Since there is a trust relationship among proxies, they can be relied upon to exchange trusted billing information pertaining to the parties involved in a call.

For these reasons, it is appropriate to consider defining SIP header extensions to allow proxies to exchange information during call setup. It is the intent that the extensions would only appear on trusted network segments, should be inserted upon entering a trusted network region, and removed before leaving trusted network segments. Rules for inserting and removing headers exchanged only between proxies are for further study.

Significant amounts of information is retrieved by an originating proxy in its handling of a connection setup request from a user agent. Such information includes location information about the subscriber (essential for emergency services calls), billing information, and station information (e.g. coin operated phone). In addition, while translating the destination number, information such as the local-number-portability office code is obtained and will be needed by all other proxies handling this call.

For Usage Accounting records, it is necessary to have an identifier that can be associated with all the event records produced for the call. Call-ID cannot be used as such an identifier since it is selected by the originating user agent, and may not be unique among all past calls as well as current calls. Further, since this identifier is to be used by the service provider, it should be chosen in a manner and in a format that meets the service provider's needs.

Billing information may not necessarily be unique for each user (consider the case of calls from an office all billed to the same account). Billing information may not necessarily be identical for all calls made by a single user (consider prepaid calls, credit card calls, collect calls, etc). It is therefore necessary to carry billing information separate from the calling and called party identification. Furthermore, some billing models call for split-charging where multiple entities are billed for portions of the call.

The addition of two SIP General Header Fields allows for the capture of billing information and billing identification for the duration of the call. Alternative techniques such as multi-part attachments will not coexist with encrypted messages.

It is the intent that the billing extensions would only appear on trusted network segments, and MAY be inserted by a proxy in INVITE requests entering a trusted network segment, and removed before leaving trusted network segments.The Dcs-Billing-ID and Dcs-Billing-Info header extensions are used only on requests and responses between proxies. They are never sent to, nor sent by, an untrusted UAC/UAS.

3.3.9.1 Syntax

The Dcs-Billing-ID and Dcs-Billing-Info headers are defined by the following BNF.

Dcs-Billing-ID = "Dcs-Billing-ID" ":" Billing-Correlation-ID

Acct-Data = 1*unreserved | (1*unreserved "," Acct-Data)
Acct-Entry = Acct-Charge-Number "/" Acct-Calling-Number "/"
Acct-Called-Number ["/" Acct-Routing-Number
"/" Acct-Location-Routing-Number]
Acct-Charge-Number = 1*unreserved
Acct-Calling-Number = 1*unreserved
Acct-Called-Number = 1*unreserved
Acct-Routing-Number = 1*unreserved
Acct-Location-Routing-Number = 1*unreserved
Billing-Correlation-ID = 1*unreserved

The Dcs-billing-ID extension contains an identifier that can be used by an event recorder to associate multiple usage records, possibly from different sources, with a billable account. Dcs-billing-id is chosen to be globally unique within the system for a window of several months. This header is only used between proxies.

The Billing-Correlation-ID is specified in [3] as a 16-byte binary structure, containing 4 bytes of NNTP timestamp, 8 bytes of MAC address of the network element that generated the ID, and 4 bytes of monotonically increasing sequence number at that network element. This MUST be encoded in the Dcs-Billing-ID header as a 32-byte hex string.

The Dcs-billing-info extension identifies a subscriber account number of the payer, and other information necessary for accurate billing of the service.

The hostport, if present, specifies a record keeping server for event messages relating to this call. If not present, the default record keeping server for each network element is sent the event messages.

Acct-data contains the information needed by the Gate Controller to give to the CMTS for generation of event message records, as specified in [3]. Acct-Charge-Number, Acct-Calling-Number, Acct-Called-Number, Acct-Routing-Number, and Acct-Location-Routing-Number are each defined as 20-byte E.164 formatted addresses.

3.3.9.2 Procedures at an Untrusted User Agent Client (UAC)

This header is never sent to an untrusted UAC, and is never sent by an untrusted UAC.

3.3.9.3 Procedures at a Trusted User Agent Client (UAC)

The UAC MUST generate the Billing-Correlation-ID for the call, and insert the Dcs-Billing-ID header into the initial INVITE message sent to the terminating proxy.

If the response to the initial INVITE is a 3xx-Redirect, the UAC generates a new initial INVITE request to the destination specified in the Contact: header, as per standard SIP[11]. If a UAC receives a 3xx-Redirect response to an initial INVITE, the INVITE generated by the UAC MUST contain the Dcs-Billing-Info headers from the 3xx-Redirect response.

An originating proxy that includes a Dcs-Also header in an initial INVITE request MUST include a Dcs-Billing-Info header in the Dcs-Also's URL. This Dcs-Billing-Info header MUST include the accounting information of the initiator.

A UAC that sends a mid-call INVITE request including a Dcs-Also header MUST include a Dcs-Billing-ID header and one or more Dcs-Billing-Info headers attached to the Dcs-Also. The Dcs-Billing-Info headers MUST include the complete set of Dcs-Billing-Info headers associated with the current call, and MUST include one additional Dcs-Billing-Info header (for the segment from the initiator) with accounting information of the initiator.

3.3.9.4 Procedures at an Untrusted User Agent Server (UAS)

This header is never sent to an untrusted UAS, and is never sent by an untrusted UAS.

3.3.9.5 Procedures at a Trusted User Agent Server (UAS)

The UAS MAY include a Dcs-Billing-Info header in the first non-100 response to an initial INVITE message if it wishes to override the billing information that was present in the INVITE (e.g. for a toll-free call). The decision to do this and the contents of the resulting Dcs-Billing-Info header MUST be determined by service provider policy provisioned in the UAS.

The UAS MUST add Dcs-Billing-Info headers to a 3xx-redirect response to an initial INVITE. All Dcs-Billing-Info headers present in the initial INVITE MUST be copied to the 3xx-redirect response. In addition, the UAS MUST add an additional Dcs-Billing-Info header, for the segment from the destination to the forwarded-to destination, giving the accounting information for the call forwarder.

3.3.9.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

For purposes of mid-call changes, such as call transfers, the proxy that receives the request from a non-trusted endpoint is considered the initiating proxy; the proxy that sends the request to a non-trusted endpoint is considered the recipient proxy. Procedures for the initiating proxy are included below with those for originating proxies, while procedures for the recipient proxy are included with those for terminating proxies.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to a non-trusted endpoint, does not generate Dcs-Billing-ID nor Dcs-Billing-Info headers.

A proxy that is neither an originating proxy nor a terminating proxy has no function in manipulating existing calls.

3.3.9.6.1 Procedures at Originating Proxy

The originating proxy MUST generate the Billing-Correlation-ID for the call, and insert the Dcs-Billing-ID header into the initial INVITE message sent to the terminating proxy.

If the Request-URI contains a private-param, and the decoded username contains billing information, the originating proxy MUST generate a Dcs-Billing-Info header with that decrypted information. Otherwise, the originating proxy MUST determine the accounting information for the call originator, and insert a Dcs-Billing-Info header including that information.

If the response to the initial INVITE is a 3xx-Redirect, received prior to a 18x-Ringing, the originating proxy generates a new initial INVITE request to the destination specified in the Contact: header, as per standard SIP[11]. If an originating proxy receives a 3xx-Redirect response to an initial INVITE prior to a 18x-Ringing response, the INVITE generated by the proxy MUST contain the Dcs-Billing-Info headers from the 3xx-Redirect response.

If the response to the initial INVITE is a 3xx-Redirect, received after a 18x-Ringing, the originating proxy generates a private URL and places it in the Contact header of a 3xx-Redirect response sent to the originating endpoint. This private URL MUST contain the sequence of Dcs-Billing-Info values, which indicate the complex charging arrangement for the new call, and an expiration time very shortly in the future, to limit the ability of the originator to re-use this private-param for multiple calls.

An originating proxy that includes a Dcs-Also header in an initial INVITE request MUST include a Dcs-Billing-Info header in the Dcs-Also's URL. This Dcs-Billing-Info header MUST include the accounting information of the initiator.

An initiating proxy that sends a mid-call INVITE request including a Dcs-Also header MUST include a Dcs-Billing-ID header and one or more Dcs-Billing-Info headers in the Dcs-Also's URL. The Dcs-Billing-Info headers MUST include the complete set of Dcs-Billing-Info headers associated with the current call, and MUST include one additional Dcs-Billing-Info header (for the segment from the initiator) with accounting information of the initiator.

3.3.9.6.2 Procedures at Terminating Proxy

The terminating proxy MUST NOT send the Dcs-Billing-ID nor the Dcs-Billing-Info headers to a non-trusted destination.

The terminating proxy MAY include a Dcs-Billing-Info header in the first non-100 response to an initial INVITE message if it wishes to override the billing information that was present in the INVITE (e.g. for a toll-free call). The decision to do this and the contents of the resulting Dcs-Billing-Info header MUST be determined by service provider policy provisioned in the terminating proxy.

The terminating proxy MUST add Dcs-Billing-Info headers to a 3xx-redirect response to an initial INVITE. All Dcs-Billing-Info headers present in the initial INVITE MUST be copied to the 3xx-redirect response. In addition, the terminating proxy MUST add an additional Dcs-Billing-Info header, for the segment from the destination to the forwarded-to destination, giving the accounting information for the call forwarder.

A proxy receiving a mid-call INVITE request that includes a Dcs-Also header generates a private URL and places it in the Dcs-Also header sent to the endpoint. This private URL MUST contain the value of Dcs-Billing-ID, the sequence of Dcs-Billing-Info values, which indicate the complex charging arrangement for the new call, and an expiration time very shortly in the future, to limit the ability of the endpoint to re-use this private-param for multiple calls.

3.3.10 DCS-LAES and DCS-REDIRECT

The Dcs-Laes extension contains the information needed to support Lawfully Authorized Electronic Surveillance [10]. This header contains the address and port of an Electronic Surveillance Delivery Function for delivery of a duplicate stream of event messages retlated to this call. The header may also contain an additional address and port for delivery of call content. Security key information is included to enable pairs of Delivery Functions to securely exchange surveillance information. This header is only used between proxies.

The Dcs-Redirect extension contains call identifying information needed to support the requirements of Lawfully Authorized Electronic Surveillance of redirected calls. This header is only used between proxies.

3.3.10.1 Syntax

The format of the Dcs-Laes header is given by the following BNF.

Dcs-LAES	**= "Dcs-LAES" ":" Laes-sig ["," Laes-content] ";" Laes-key**
Laes-sig	**= hostport**
Laes-content	**= hostport**
Laes-key	**= token**
Dcs-Redirect	**= "Dcs-Redirect" ":" Called-id Redirector Num-redir**
Called-id	**= "<" SIP-URL ">"**
Redirector	**= "<" SIP-URL ">"**
Num-redir	**= 1*DIGIT**

The values of Laes-sig and Laes-content are addresses of the Electronic Surveillance Delivery Function, and used as the destination address for call-identifying information and call-content, respectively.

Laes-key is a string generated by the proxy that is used by the Delivery Function to securely transfer information between them.

3.3.10.2 Procedures at an Untrusted User Agent Client (UAC)

This header is never sent to an untrusted UAC, and is never sent by an untrusted UAC.

3.3.10.3 Procedures at a Trusted User Agent Client (UAC)

The UAC checks for an outstanding lawfully authorized surveillance order for the originating subscriber, and, if present, includes this information in the Authorization for Quality of Service or signals this information to the device performing the intercept (e.g. a Media Gateway).

If the Dcs-LAES header is present in the 183-Session-Progress response (indicating surveillance is required on the terminating subscriber, but that the terminating equipment is unable to perform that function), the UAC MUST include this information in the Authorization for Quality of Service, or MUST signal this information to the device performing the intercept (e.g. a Media Gateway).

If a 3xx-Redirect response is received to the initial INVITE request, and if a Dcs-LAES header is present in the 3xx response, the UAC MUST include that header unchanged in the reissued INVITE. The UAC MUST also include a Dcs-Redirect header containing the original dialed number, the new destination number, and the number of redirections that have occurred.

A UAC that includes a Dcs-Also header in an INVITE request, when the originating subscriber has an outstanding lawfully authorized surveillance order, MUST include a Dcs-Laes header attached to the Dcs-Also. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be intercepted, and MUST include a random string for use as a security key between the Delivery Functions.

3.3.10.4 Procedures at an Untrusted User Agent Server (UAS)

This header is never sent to an untrusted UAS, and is never sent by an untrusted UAS.

3.3.10.5 Procedures at a Trusted User Agent Server (UAS)

The UAS checks for an outstanding lawfully authorized surveillance order for the terminating subscriber. If present, the UAS includes this information in the authorization for Quality of Service.

If the terminating equipment is unable to perform the required surveillance (e.g. if the destination is a voicemail server), the UAS MUST include a Dcs-LAES header in the 183-Session-Progress response requesting the originating proxy to perform the surveillance. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be intercepted, and MUST include a random string for use as a security key between the Delivery Functions.

If the response to the initial INVITE request is a 3xx-Redirect response, and there is an outstanding lawfully authorized surveillance order for the terminating subscriber, the UAS MUST include a Dcs-Laes header in the 3xx-Redirect response, with contents as described above.

3.3.10.6 Procedures at Proxy

Two sets of proxy procedures are defined: (1) the procedures at an originating proxy, and (2) the procedures at a terminating proxy.

The originating proxy is a proxy that received the INVITE request from a non-trusted endpoint.

The terminating proxy is a proxy that sends the INVITE request to a non-trusted endpoint.

For purposes of mid-call changes, such as call transfers, the proxy that receives the request from a non-trusted endpoint is considered the initiating proxy; the proxy that sends the request to a non-trusted endpoint is considered the recipient proxy. Procedures for the initiating proxy are included below with those for originating proxies, while procedures for the recipient proxy are included with those for terminating proxies.

A proxy that both receives the INVITE request from an untrusted endpoint, and sends the INVITE request to a non-trusted endpoint, does not generate Dcs-Laes nor Dcs-Redirect headers.

A proxy that is neither an originating proxy nor a terminating proxy has no function in manipulating existing calls.

3.3.10.6.1 Procedures at Originating Proxy

The originating proxy checks for an outstanding lawfully authorized surveillance order for the originating subscriber, and, if present, includes this information in the Authorization for Quality of Service or signals this information to the device performing the intercept (e.g. a Media Gateway).

If the Dcs-LAES header is present in the 183-Session-Progress response (indicating surveillance is required on the terminating subscriber, but that the terminating equipment is unable to perform that function), the originating proxy MUST include this information in the Authorization for Quality of Service, or MUST signal this information to the device performing the intercept (e.g. a Media Gateway).

If the Request-URI in an initial INVITE request contains the private-param user parameter, the originating proxy MUST decrypt the username information to find the real destination for the call, and other special processing information. If electronic surveillance information is contained in the decrypted username, the originating proxy MUST generate a Dcs-LAES header with the surveillance information.

If a 3xx-Redirect response is received to the initial INVITE request prior to a 18x-Ringing, and if a Dcs-LAES header is present in the 3xx response, the originating proxy MUST include that header unchanged in the reissued INVITE. The originating proxy MUST also include a Dcs-Redirect header containing the original dialed number, the new destination number, and the number of redirections that have occurred.

If a 3xx-Redirect response is received to the initial INVITE request after a 18x-Ringing, the originating proxy generates a private URL and places it in the Contact header of a 3xx-Redirect response sent to the originating endpoint. If a Dcs-Laes header is present in the 3xx response, this private URL MUST contain (1) the electronic surveillance information from the 3xx-Redirect response, (2) the original destination number, (3) the identity of the redirecting party, and (4) the number of redirections of this call.

An originating proxy that includes a Dcs-Also header in an initial INVITE request, when the originating subscriber has an outstanding lawfully authorized surveillance order, MUST include a Dcs-Laes header in the Dcs-Also's URL. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be intercepted, and MUST include a random string for use as a security key between the Delivery Functions.

An initiating proxy that sends a mid-call INVITE request including a Dcs-Also header, when the initiating subscriber has an outstanding lawfully authorized surveillance order, MUST include a Dcs-Laes header in the Dcs-Also's URL. The Dcs-Laes header MUST include the information listed above.

3.3.10.6.2 Procedures at Terminating Proxy

The terminating proxy checks for an outstanding lawfully authorized surveillance order for the terminating subscriber. If present, the terminating proxy includes this information in the authorization for Quality of Service.

The terminating proxy MUST NOT send the Dcs-Laes and Dcs-Redirect headers to an untrusted endpoint.

If the terminating equipment is unable to perform the required surveillance (e.g. if the destination is a voicemail server), the terminating proxy MUST include a Dcs-LAES header in the 183-Session-Progress response requesting the originating proxy to perform the surveillance. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be intercepted, and MUST include a random string for use as a security key between the Delivery Functions.

If the response to the initial INVITE request is a 3xx-Redirect response, and there is an outstanding lawfully authorized surveillance order for the terminating subscriber, the terminating proxy MUST include a Dcs-Laes header in the 3xx-Redirect response, with contents as described above.

A proxy receiving a mid-call INVITE request that includes a Dcs-Also header with a Dcs-laes header attached MUST generate a private URL and place it in the Dcs-Also header sent to the endpoint. This private URL MUST contain the Dcs-Laes information from the attached header.

3.3.11 Session

The Session header is contained in a 183-Session-Progress response (see section 3.4.1), and indicates the reason for the response.

When combined with an INVITE containing SDP preconditions for QoS and/or security, it provides a simple mechanism for establishing quality of service and security between session clients prior to alerting the user.

Additionally, the Session header contained in a 183-Session-Progress response provides a mechanism for establishing a media path (for e.g. call progress tones, or recorded announcements) prioir to full session setup.

The applications of this header are described more fully in section 3.4.1. This extension is more fully described in [16].

3.3.11.1 Syntax

The BNF description of this header is:

Session = "Session" ":" 1#session-tag

session-tag = "Media" | "QoS" | "Security"

The Session header will be used to communicate to the calling user agent the reason for a SDP message body being included in an 18x message. The valid values for the Session header are Media, QoS and Security. Multiple of these values can be indicated.

A value of "Media" indicates that the SDP SHOULD be used for establishing an early media session. The early media session will generally be used to communicate the status of the session but could also be used for other reasons. For instance, it could be used to play music while the calling user is being alerted.

A value of "QoS" indicates that the SDP SHOULD be used for establishing a QoS relationship between the calling and called user agents. This could involve the user agents requesting QoS resources using RSVP or some other signaling mechanism.

A value of "Security" indicates that the SDP SHOULD be used for establishing a security relationship between the calling and called user agents. This could be an IPSEC based relationship, for example.

3.3.11.2 Procedures at an Untrusted User Agent Client (UAC)

When the UAC receives a 183 response that contains a session description and a Session header that indicates "qos" or "security," it MUST perform the resource reservation and/or security establishment as described in section 3.2.2.

When the UAC receives a 183 response that contains a session description and a Session header that indicates "media," it MUST setup the associated media session and present any media received from the UAS to the user.

3.3.11.3 Procedures at a Trusted User Agent Client (UAC)

When the UAC receives a 183 response that contains a session description and a Session header that indicates "qos" or "security," it MUST perform the resource reservation and/or security establishment as described in section 3.2.2.

When the UAC receives a 183 response that contains a session description and a Session header that indicates "media," it MUST setup the associated media session and present any media received from the UAS to the user.

3.3.11.4 Procedures at an Untrusted User Agent Server (UAS)

When the UAS receives an INVITE request that contained a SDP X-pc-qos or X-pc-security parameter, the UAS MUST send a 183-Session-Progress provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates "qos" and/or "security."

3.3.11.5 Procedures at a Trusted User Agent Server (UAS)

When the UAS receives an INVITE request that contained a SDP X-pc-qos or X-pc-security parameter, the UAS MUST send a 183-Session-Progress provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates "qos" and/or "security."

When the UAS receives an INVITE request that results in the need to report on the status of the media setup through a media stream, the UAS MAY send a 183 provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates "media."

3.3.11.6 Procedures at Proxy

The Session header is ignored by proxies.

3.3.12 RSEQ and RACK

The RSeq and Rack headers, combined with the PRACK method (section 3.2.1), provides a simple extension to SIP for ensuring that provisional responses to all SIP requests are delivered reliably end to end, independent of the underlying transport mechanism. The extension works for provisional responses for any method. The extension is simple, requiring two new header fields, and one new method. The extension does not require support in proxies. The extension is indicated with the option tag org.ietf.sip.100rel. Further information about this extension is contained in [25]

3.3.12.1 Syntax

The BNF description of the RSeq and RAck headers are as follows:

RSeq	**= "RSeq" ":" response-num**
RAck	**= "RAck" ":" response-num Cseq-num Method**
Response-num	**= 1*digit**
Cseq-num	**= 1*digit**

The RSeq number in any reliable provisional response MUST be between 1 and 2**32 - 1. The value in the first reliable provisional response is randomly chosen by the UAS. It MUST be between 1 and 2**31 - 1. It is RECOMMENDED that it be chosen uniformly in this range. The RSeq numbering space is within a single request. This means that provisional responses for different requests MAY use the same values for the RSeq number. Reliable provisional responses for the same request MUST contain RSeq values which increment by exactly one for each response. RSeq numbers MUST NOT wrap around. Because the initial one is chosen to be less than 2**31 - 1, but the maximum is 2**32 - 1, there can be up to 2**31 reliable provisional responses per request, which is more than sufficient.

The RAck header contains two numbers and a method tag. The first number is the value from the RSeq header in the provisional response that is being acknowledged. The next number, and the method, are copied from the CSeq in the response that is being acknowledged.

3.3.12.2 Procedures at User Agent Client (UAC)

The procedures at the client are described in section 3.2.1.

3.3.12.3 Procedures at User Agent Server (UAS)

The procedures at the server are described in section 3.2.1.

3.3.12.4 Procedures at Proxy

The RSeq and RAck headers are ignored by proxies.

3.4 SIP Response Extensions

3.4.1 183 Session Progress

There are instances, most notably dealing with SIP to PSTN interworking, that necessitate that the SIP UAS be able to suppress local alerting by the UAC and to set up a preliminary media session from the UAS to the UAC. This would allow the UAS to play back media prior to the full SIP session being set up. This media would be used to report on the status of the session setup request. It could also be used to play music while the session setup is attempted. This would be useful for find-me like services that involve attempting multiple locations for a single setup request.

The only method in the current SIP specification that allows the UAS to playback media is to set up a full SIP session. In PSTN interworking situations (and likely in end-to-end SIP sessions) this will cause a billing relationship to be established between networks for the session. This causes a problem when the reason for setting up the media session is to indicate a failure in the session setup.

To allow for transmission of temporary media which does not correspond to the four provisional status codes defined in SIP [11], this protocol extension defines one additional response code of "183 Session Progress."

The 183 Session Progress response can be used for any arbitrary inband communication of call status. It is not, however, used to convey ringing, forwarding, or call queueing situations.

The format of this provisional response is identical to that of 200-class responses to INVITE requests. Under most circumstances, provisional responses used to initiate temporary media will contain SDP that is a subset of the media description presented in the INVITE message (as in normal 200 responses). The media streams will be established after the message confirming receipt of the provisional response has been sent (from the client's perspective) or received (from the server's perspective).

The 183 Session Progress may be sent to communicate the status of the session setup attempt as part of a media stream. The called user agent will indicate this by including the Session header with a value of media.

In this case, the calling UA establishes a media session according to the contents of the session description contained in the 183 message. The calling UA does not apply local alerting that would interfere with the media session information supplied by the called UA.

The 183 message includes enough session description information to allow for a media session between the called UA and the calling UA.

A 183 response to an INVITE indicates the destination will generate a ringback audio stream that is to be played to the call originator. This response is only generated by trusted network entities.

The designation of media capabilities in a provisional response has no implications on the capabilities of any subsequent temporary connections or the final connection. Each media stream is negotiated relative to the session description in the original INVITE request.

Sending of temporary media MUST be discontinued upon the sending (from the server's perspective) or the receipt (from the client's perspective) of any INVITE final response.

Use of this header is further described in [16]

3.4.1.1 Procedures at an Untrusted User Agent Client (UAC)

When the UAC receives a 183 response that contains a session description and a Session header that indicates “qos” or “security,” it MUST perform the resource reservation and/or security establishment as described in section 3.2.2.

When the UAC receives a 183 response that contains a session description and a Session header that indicates “media,” it MUST setup the associated media session and present any media received from the UAS to the user.

3.4.1.2 Procedures at a Trusted User Agent Client (UAC)

When the UAC receives a 183 response that contains a session description and a Session header that indicates “qos” or “security,” it MUST perform the resource reservation and/or security establishment as described in section 3.2.2.

When the UAC receives a 183 response that contains a session description and a Session header that indicates “media,” it MUST setup the associated media session and present any media received from the UAS to the user.

3.4.1.3 Procedures at an Untrusted User Agent Server (UAS)

When the UAS receives an INVITE request that contained a SDP X-pc-qos or X-pc-security parameter, the UAS MUST send a 183-Session-Progress provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates “qos” and/or “security.”

3.4.1.4 Procedures at a Trusted User Agent Server (UAS)

When the UAS receives an INVITE request that contained a SDP X-pc-qos or X-pc-security parameter, the UAS MUST send a 183-Session-Progress provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates “qos” and/or “security.”

When the UAS receives an INVITE request that results in the need to report on the status of the media setup through a media stream, the UAS MAY send a 183 provisional response to the UAC; the 183-Session-Progress provisional response MUST contain a message body and MUST contain a Session header that indicates “media.”

3.4.1.5 Procedures at Proxy

Provisional responses of type 183 MUST be forwarded.

3.4.2 580 Precondition Failure

The 580-Precondition-Failure is a server failure error code. It is sent by the UAS as a final response to an INVITE request that specified mandatory qos and/or security preconditions for the session. If those preconditions were unable to be satisfied, the UAS responds with the 580-Precondition-failure error code.

For further description, see section 3.2.2.

3.4.2.1 Procedures at an Untrusted User Agent Client (UAC)

The procedures at the client are described in section 3.2.2.

3.4.2.2 Procedures at a Trusted User Agent Client (UAC)

The procedures at the client are described in section 3.2.2.

3.4.2.3 Procedures at an Untrusted User Agent Server (UAS)

The procedures at the server are described in section 3.2.2.

3.4.2.4 Procedures at a Trusted User Agent Server (UAS)

The procedures at the server are described in section 3.2.2.

3.4.2.5 Procedures at Proxy

Proxies MUST handle the 580-Precondition-Failure error response identical to all other 5xx error responses.

4. SIP Profile

This section defines a SIP [11] profile for usage in DCS compatible systems. This section is structured to mirror the SIP document and its section numbering. The subsections of this section are numbered such that the second digit tracks the SIP section numbers of RFC2543, and section titles at all header levels track RFC2543.

This section, and section 3 preceding, define the nearly complete set of enhancements and restrictions to a standard SIP implementation based on RFC2543/RFC2543bis. However, not all details of the required behavior can be captured in these sections. Later sections provide details needed for certification and interoperability testing, which are generally not present in RFC2543. Sections 3 through 11 are considered normative. Appendicies are provided to give informative examples of the use of SIP in achieving the services listed in Section 2.

4.1 Introduction

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 1 except as defined in this section.

4.1.1 Overview of SIP Operations

The following sections define the overview of SIP operation as it applies to DCS compliant applications.

4.1.1.1 Locating a SIP Server

DCS replaces standard SIP proxy servers with CMS/Proxies. CMS/Proxies play a critical role in ensuring only authorized users have access to the QoS features of DCS. As such, the MTA's associated CMS/Proxy location MUST be provisioned in the MTA.

4.1.1.2 SIP Transaction

DCS is a signaling specification designed for use in real-time telephone applications. Users of the existing telephone network have come to expect a certain level of real-time performance when placing a call. The real-time performance parameters include low post-dial delay and low post-pickup delay. Using TCP as a reliable transport mechanism for DCS signaling is impractical because TCP is not ideal for use as a real-time transport protocol. As such, all transaction initiated by DCS clients and proxies MUST use UDP [14]. DCS clients and proxies MAY also be capable of supporting transactions sent to it using TCP [15].

4.1.1.3 SIP Invitation

To support CODEC selection, an SDP session description MUST be included in:

1) the INVITE request,
2) the 183-Session-Progress provisional response to the INVITE, and
3) the PRACK acknowledging the 183-Session-Progress provisional response

4.1.1.4 Locating a User

Proxies are responsible for locating users within the system. A proxy which forwards SIP requests MUST add itself to the beginning of the list of forwarders noted in the Via headers.

DCS is designed to ensure originating MTA location privacy. As such location information about the originating MTA is not delivered to the terminating MTA. SIP requires proxies to insert "Via" headers in support of response reverse routing and SIP allows encryption of request "Via" headers in support of privacy. As such, CMS/proxies MUST encrypt all but the top most "Via" header of a request to a meaningless string in support of user privacy. CMS/proxies that encrypt request "Via" headers MUST restore the unencrypted "Via" headers in the response.

4.1.1.5 Changing an Existing Session

DCS supports changing CODECs at any time during a session. This is accomplished by sending an INVITE that includes a new SDP session description. The re-INVITE MUST have a higher "CSeq" than any previous request from the client to the server. Endpoints authorized for a bandwidth lower than that required by the new SDP session description MUST send the re-INVITE on the proxy-proxy signaling path. Endpoints authorized for bandwidth higher that that required by the new SDP session description MAY send the re-INVITE on the end-end signaling path. See section 6 and 7.1.1 for definition of the proxy-proxy and end-end signaling paths.

4.1.1.6 Registration Services

Proxies MUST support SIP REGISTER request in support of call forwarding features.

4.1.2 Protocol Properties

4.1.2.1 Minimal State

Proper operation of the DCS QoS and privacy features require an MTA transaction with the CMS/proxy for each new SIP session. As such, MTAs SHOULD NOT cache other MTA locations.

4.2 SIP Uniform Resource Locators

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 2 except as defined in this section.

DCS defines extensions to the SIP-URL; this specification refers to such as a DCS-URL. The DCS-URL is syntactically compatible to the SIP-URL defined in [11] section 2, Figure 3. The DCS-URL MUST be as given in section 3.1.

4.3 SIP Message Overview

DCS messages MUST follow the requirements defined in [11] section 3 except as defined in this section.

As explained in 4.1.1.2, DCS MUST use UDP for message exchanges between proxies and between MTAs and its proxy.

4.4 Request

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 4 except as defined in this section.

4.4.1 Request-Line

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 4.1 and the Request-URI MUST use the DCS-URL syntax defined in section 3.1.

4.4.2 Request-URI

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 4.3 except as defined in this section.

The Request-URI is a DCS-URL, as defined in section 3.1, or a tel: URL as defined in [18].

The request line of an INVITE, associated with a basic call, MUST identify the user using a tel: URL or by using the telephone-subscriber syntax (i.e. the dialed phone number) in a sip: URL. When used with a sip: URL, it MUST identify the host as the CMS or endpoint to which the message is addressed. Other request lines associated with a basic call MUST identify the host using IPv4address or FQDN syntax, as given by the contact header.

The Request-URI of an INVITE associated with a special service requested by an untrusted UAC MAY use the private-param url-parameter. Examples include call-forwarding, call-return, and call-transfer. When identified by the hostname of the URL, the proxy MUST decode/decrypt the username, and replace the URL with a new DCS-URL. If the replacement DCS-URL contains a private-param, it MUST identify a different hostname than the current proxy. If the proxy is unable to decode/decrypt the username, it MUST reject the call attempt with an appropriate 4xx error code.

The host part of the Request-URI typically agrees with one of the host names of the receiving server. However, if the Request-URI of an INVITE received at a proxy from an untrusted UAC does not, the server SHOULD proxy the request to a server or agent based on saved translation information or pre-provisioned policy information. Typically, this occurs for mid-call changes, and the saved translation information is available in a Dcs-State header attached to the INVITE request.

The Request-URI of an INVITE request sent from a proxy to another proxy MAY identify the user using the LNP information. If the proxy determines the dialed number is a LNP number, the originating proxy modifies the original request line URL by adding the LNP information and transmits the modified request line to the terminating proxy. When the request is received at the terminating proxy, the terminating proxy removes the LNP information and transmits the modified request line to the UAS.

The Request-URI of a REGISTER request typically identifies the proxy using the IPv4address syntax, but MAY be system dependent.

4.5 Response

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 5.

4.6 Header Field Definitions

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 6 except as defined in this section.

The following SIP headers MUST supported by DCS compliant applications.

1) Call-ID
2) Contact
3) CSeq
4) Expires
5) Record-Route
6) From
7) To
8) Via
9) Content-Length
10) Content-Type
11) Route
12) Require
13) Proxy-Require

The following SIP headers MAY be optionally supported by DCS compliant applications. DCS compliant applications SHOULD ignore unsupported optional headers.

1) Accept
2) Accept-Encoding
3) Accept-Language
4) Date
5) Encryption
6) Timestamp
7) Content-Encoding
8) Authorization
9) Hide
10) Max-Forwards
11) Organization
12) Priority
13) Proxy-Authorization
14) Response-Key
15) Subject
16) User-Agent
17) Allow
18) Proxy-Authenticate
19) Retry-After
20) Server
21) Unsupported
22) Warning
23) WWW-Authenticate

4.6.1 Call-ID

DCS restricts the “Call-ID” header in support of user privacy.

When anonymity is requested by the call originator, the “host” in the Call-ID MUST be “localhost”, and ”local-id” MUST be a random identifier, and SHOULD be unique across all possible MTAs with probability of greater than 0.999999. A suggested implementation is a text encoding of a cryptographic hash of phone number, time, a random number, and a quantity provisioned or manufactured to be unique across MTAs of otherwise identical manufacture. The last quantity is suggested to help prevent MTAs of an otherwise identical manufacture from producing identical “random” Call-Ids when presented with identical stimuli.

4.6.2 Contact

The “Contact” header MUST appear in the initial INVITE request, and MUST appear in the 183-Session-Progress response. This is needed to support the direct end-to-end signaling path used by most requests after the initial INVITE. The “Contact” header in an INVITE and in the 183-Session-Progress response MUST identify the host using the IPv4address or FQDN syntax.

The “Contact” header MUST appear in a 3xx response to an initial INVITE generated by a UAS, and SHOULD identify the new address with a tel: URL.

The "Contact" header when given to an untrusted UAC in a 3xx response MUST contain a private-URL, as described in 3.1, generated by a trusted proxy. To generate the username of a private URL, the proxy includes (1) the initial URL, (2) the Dcs-Billing-Info header(s) for the desired new call indicating the forwarding party is paying for part of the new call, (3) the Dcs-Billing-ID for the desired new call, (4) the Dcs-Redirect information for the desired new call, (5) an expiration time beyond which the URL is useless, MAY contain (6) any other information the proxy desires, and (7) sufficient checksum information to prevent tampering by the untrusted endpoint. This information is encoded or encrypted such that the endpoint is unable to discern the initial URL. The string is encrypted with a symmetric privately-held key, and converted to a printable string using Base64 encoding. The proxy identifies itself in the hostname of the private URL.

4.6.3 Content-Length

The "Content-Length" MUST be present when a message body is attached to the request or response. See section 4.8.2.

4.6.4 Content-Type

The "Content-Type" header, when present, MUST indicate "application/sdp". See section 4.8.1.

4.6.5 Expires

The "Expires" header MUST appear in REGISTER requests, MAY be ignored in INVITE requests, and MAY appear in 302-Redirect responses.

4.6.6 From

In support of user privacy, DCS restricts the allowable contents of the SIP "From:" header.

When the call originator requests anonymity (e.g. Dcs-Anonymity containing Name, URL, or IPAddr), compliant applications MUST generate a From: header according to the following rules:

1. The display-name MUST be absent.
2. The addr-spec MUST contain a random identifier for username, which MUST be regenerated for each call.
3. The addr-spec MUST contain the non-identifying hostname "localhost".

When both From: and To: contain random identifiers, they MUST NOT be equal.

4.6.7 Proxy-Require

The "Proxy-Require" header MUST appear in requests sent by a user agent client and the option-tag MUST be com.PacketCable.dcs.i01. It is hoped that future versions of SIP will incorporate most, if not all, of the DCS extensions and that the Proxy-Require header will not be necessary in future versions of DCS. As such, the Proxy-Require header is not shown in the message format requirements or call flow examples.

4.6.8 Require

The "Require" header MUST appear in requests sent by a user agent client and the option-tag MUST be com.PacketCable.dcs.i01. It is hoped that future versions of SIP will incorporate most, if not all, of the

DCS extensions and that the Require header will not be necessary in future versions of DCS. As such, the Require header is not shown in the message format requirements or call flow examples.

4.6.9 To

In support of user privacy, DCS restricts the allowable contents of the SIP "To:" header. While a typical To: header might contain the sequence of dialed digits used to initiate the call, this information is of end-to-end significance, and might reveal information about the caller's location, e.g. local vs. long-distance vs. pbx vs. international.

When the call originator requests anonymity (e.g. Dcs-Anonymity containing Name, URL, or IPAddr), compliant applications MUST generate a To: header according to the following rules:

1. The display-name MUST be absent.
2. The addr-spec MUST contain a random identifier for username, which MUST be regenerated for each call.
3. The addr-spec MUST contain the non-identifying hostname "localhost".

When both From: and To: contain random identifiers, they MUST NOT be equal.

Typically, the "To" header indicates the dialed digits in a telephone-URI[18].

4.6.10 Via

To support user privacy, the proxy associated with an untrusted UAS terminating a connection, MUST encrypt all "Via" headers except the top most header (i.e. the "Via" header of the terminating proxy) to a non-recognizable string (as described in RFC2543 section 6.22). The proxy MAY include the encrypted string in the Via header, with attribute ";private", or MAY cache the encrypted "Via:" headers and include a local token string in the Via header (also see section 4.1.1.4).

Typically "Via" headers indicate a host using the IPv4address syntax, for two reasons. First, use of an FQDN in the Via header could require a DNS lookup while processing the response to the request, and therefore increase the latency. Second, where multiple systems share a common FQDN but have individual IP addresses (e.g. a CMS cluster), the transaction state needed to process the response is typically only stored in one of the cluster elements; the response needs to be routed to that particular element.

4.7 Status Code Definitions

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 7 except as defined in this section.

4.7.1 302 Moved Temporarily

The address given in the Contact header is valid only for this call, and MUST NOT be cached for future calls.

4.8 SIP Message Body

DCS compliant applications MUST be in accordance with SIP version 2 [11] section 8 except as defined in this section.

4.8.1 Body Inclusion

The message body MUST appear in an INVITE request that does not have a call-leg identification matching an existing call.

An INVITE request that contains a call-leg identification matching an existing call MUST either contain a message body (e.g. indicating a hold, resume, or CODEC change) or contain a Dcs-Also or Dcs-Replaces header (e.g. indicating a call control operation) or contain a Dcs-OSPS header (e.g. indicating an emergency interrupt).

The message body MUST appear in the 183-Session-Progress response when the session: header indicates "qos" or "security", and a message body MUST appear in the PRACK message acknowledging this message.

The message body MUST appear in the 200-OK response to an INVITE that performs a hold or resume function.

All other requests and responses MUST NOT contain a message body.

The message body MUST of type "application/sdp".

4.8.2 Message Body Length

The body length in bytes MUST be given by the Content-length header field.

4.9 Compact Form

DCS compliant applications MUST support short and long form field names as defined in [11] section 9.

4.10 Behavior of SIP Clients and Servers

Behavior of DCS clients (MTAs) and servers (proxies) MUST be in accordance with section 6 and section 7 of this document.

4.10.1 General Remarks

4.10.1.1 Requests

If a user agent receives a request with a Call-ID that matches an in-progress call, but the comparison with the From header or the To header do not match, the user agent SHOULD reject the request. Such a request would require the user agent to establish a local conference bridge, which is not required in DCS.

4.10.1.2 Responses

100 responses SHOULD NOT be forwarded, other 1xx responses MUST be forwarded, after the server eliminates responses with status codes that had already been sent earlier.

4.11 Behavior of SIP User Agents

Behavior of DCS User Agents (MTAs) and CMS/Agents MUST be in accordance with section 6 and section 7 of this document.

4.11.1 Caller Issues Initial INVITE Request

The Request-URI in the request contains the address of the callee. The From and To fields in the request might contain random strings that protect the privacy of the call originator. The UAC MUST insert a Contact header into the initial INVITE request.

4.11.2 Callee Issues Response

If the UAS issues a 1xx or 2xx response to the INVITE, it MUST insert a Contact header field in the first non-100 response.

4.11.3 Caller or Callee Generate Subsequent Requests

The Contact header MUST NOT be different than the Contact header field sent in previous requests or responses.

The Request-URI MUST be set to the value of the Contact header received in the initial INVITE or response to the initial INVITE.

DCS defines two signaling paths between the UAC and UAS, called the proxy-proxy signaling path and end-end signaling path. These are shown in Figure 8 in Section 7.1.1 for the various configurations of trusted and untrusted endpoints. The proxy-proxy signaling path corresponds to the UAC/UAS sending the request to its preconfigured outbound proxy, and the end-end signaling path corresponds to the UAC/UAS sending the request to the address given in the Contact header. The initial INVITE request MUST be sent on the proxy-proxy signaling path. After generating the subsequent request message, the initiator MUST send the request on the proxy-proxy signaling path if it contains a Dcs-Also header, or if it contains a message body. All other requests SHOULD be sent on the end-end signaling path. This is consistent with the description in RFC2543bis[27].

4.12 Behavior of SIP Proxy and Redirect Servers

Behavior of proxies and redirect servers MUST be in accordance with section 6 and section 7 of this document.

4.12.1 Proxy Server

The CMS/proxy SHOULD become stateless upon sending the first non-100 response to an initial INVITE, and process the remaining provisional and final responses as a stateless proxy.

4.12.1.1 Stateful Proxy: Receiving Requests

When a stateful proxy receives a request, it checks the To, From (including tags), Call-ID, CSeq, and Request-URI against existing request records. If the tuple exists, the request is a retransmission.

4.13 Security Considerations

DCS security considerations MUST be in accordance with the PacketCable Security specification [2].

4.14 SIP Authentication using HTTP Basic and Digest Schemes

DCS authentication MUST be in accordance with section 7 and the PacketCable Security specification [2].

4.15 SIP Security Using PGP

DCS security MUST be in accordance with the PacketCable Security specification [2].

4.16 Examples

Examples of DCS messages are shown in sections 6 and 7, and in the appendices.

5. SDP Profile for use by DCS

The use of SDP in DCS is defined by the following and in the order stated:
1. The SDP profile presented below:
2. RFC 2543 [11], Appendix B (*SIP: Usage of the Session Description Protocol (SDP)*)
3. RFC 2327 [12] (*SDP: Session Description Protocol*)

1. Protocol Version (v=)
   v= <version>
   v= 0

   Send: In accordance with RFC 2327[12] (i.e. v=0)
   Receive: in accordance with RFC 2327[12].

2. Origin Consists (o=) of 6 sub-fields in RFC2327[12]:
   o= <username> <session-ID> <version> <network-type> <address-type> <address>
   o= - 2987933615 2987933615 IN IP4 A3C47F2146789F0

   2.1. Username:
   Send: Hyphen MUST be used as username when privacy is requested. Hyphen SHOULD be used otherwise.
   Receive: This field SHOULD be ignored.
   2.2. Session-ID:
   Send: MUST be in accordance with RFC 2327[12] for interoperability with non-DCS clients.
   Receive: This field SHOULD be ignored.
   2.3. Version:
   Send: In accordance with RFC 2327[12].
   Receive: This field SHOULD be ignored.
   2.4. Network Type:
   Send: Type 'IN' MUST be used.
   Receive: This field SHOULD be ignored.
   2.5. Address Type:
   Send: Type "IP4" MUST be used
   Receive: This field SHOULD be ignored.
   2.6. Address:
   Send: A value considered unique to the originator MUST be used. The Call-ID value in the SIP messages SHOULD be used. When privacy is requested, the value used MUST NOT reveal any caller information.
   Receive: This field MUST be ignored.

3. Session Name (s=)
   s= <session-name>
   s= -

   Send: Hyphen MUST be used as Session name.
   Receive: This field MUST be ignored.

4. Session and Media Information (i=)
   i= <session-description>

   Send: If privacy is requested by the application this field MUST NOT be used, otherwise it SHOULD NOT be used.
   Receive: This field MUST be ignored.

5. URI (u=)
   u= <URI>

   Send: If privacy is requested by the application this field MUST NOT be used, otherwise it SHOULD NOT be used.
   Receive: This field MUST be ignored.

6. E-Mail Address and Phone Number (e=, p=)
   e= <e-mail-address>
   p= <phone-number>

   Send: If privacy is requested by the application this field MUST NOT be used, otherwise it SHOULD NOT be used.
   Receive: This field MUST be ignored.

7. Connection Data (c=) consists of 3 sub-fields:
   c= <network-type> <address-type> <connection-address>
   c= IN IP4 10.10.111.11

   7.1. Network Type:
   Send: Type 'IN' MUST be used.
   Receive: Type "IN" MUST be present.
   7.2. Address Type:
   Send: Type "IP4" MUST be used
   Receive: Type "IP4" MUST be used
   7.3. Connection Address:
   Send: This field MUST be filled with unicast IP address at which the application will receive media stream, thus a TTL value MUST NOT be present and a "number of addresses" value MUST NOT be present. The field MUST NOT be filled with a domain name. A non-zero address specifies both the send and receive address for the media stream(s) it covers.
   Receive: A unicast IP address or a fully qualified domain name MUST be present. A non-zero address specifies both the send and receive address for the media stream(s) it covers.

8. Bandwidth (b=)
   b= <modifier> : <bandwidth-value>
   b= AS : 64

   Send: Bandwidth information is optional in SDP but it SHOULD always be included. [2] When an rtpmap or a non well-known codec[3] is used, the bandwidth modifier MUST be used.
   Receive: Bandwidth information SHOULD be included. If a bandwidth modifier is not included, the receiver MUST assume reasonable default bandwidth values for well-known codecs.

   8.1. Modifier:
   Send: Type 'AS' MUST be used.
   Receive: Type "AS" MUST be present.
   8.2. Bandwidth Value: The maximum bandwidth that will be used with the media stream.
   Send: The field MUST be filled with the Maximum Bandwidth requirement of the Media stream in kilobits per second.
   Receive: The maximum bandwidth requirement of the media stream in kilobits per second MUST be present.

[2] If this field is not used, the DCS Proxy and Gate Controller might not authorize the appropriate bandwidth.
[3] A non well-known codec is a codec not defined in the PacketCable™ codec specification [7].

9. Time, Repeat Times and Time Zones (t=, r=, z=)
   t= <start-time> <stop-time>
   t= 36124033 0
   r= <repeat-interval> <active-duration> <list-of-offsets-from-start-time>
   z= <adjustment-time> <offset>

   Send: Time MUST be present; start time SHOULD be current time, and stop time SHOULD be zero. Repeat Times, and Time Zones SHOULD NOT be used, if they are used it should be in accordance with RFC 2327[12].
   Receive: If any of these fields are present, they SHOULD be ignored.

10. Encryption Keys:
    k= <method>
    k= <method> : <encryption-keys>

    Security services for PacketCable are defined by the PacketCable Security specification [2]. The security services specified for RTP and RTCP do not comply with those of RFC 1889, RFC 1890, RFC 2327, and RFC 2543. In the interest of interoperability with non-PackietCable devices, the "k=" parameter will therefore not be used to convey security parameters.

    Send: MUST NOT be used.
    Receive: MUST be ignored.

11. Attributes (a=)
    a= <attribute> : <value>
    a= rtpmap : <payload type> <encoding name>/<clock rate>[/<encoding parameters>]
    a= rtpmap : 0 PCMU / 8000
    a= X-pc-codecs: <alternative1> <alternative2> …
    a= X-pc-csuites:<alternative1> <alternative2> …
    a= X-pc-secret: <method>:<encryption key>
    a= X-pc-secret:clear:PackMyBoxWithFiveDozenLiquorJugs

    a= X-pc-qos: <mandatory> <sendrecv> <confirm>
    a= X-pc-security: <mandatory> <sendrecv> <confirm>

    a= <attribute>
    a= recvonly
    a= sendrecv
    a= sendonly

    Send: One or more "a" attribute lines MAY be included. An attribute line not specified below SHOULD NOT be used.
    Receive: One or more of the "a" attribute lines specified below MAY be included and MUST be acted upon accordingly. "a" attribute lines not specified below may be present but MUST be ignored.

    11.1. rtpmap:
    Send: The field MUST be used in accordance with RFC 2327[12]. It MAY be used for well-known as well as well as non well-known codecs. The encoding names MUST be as defined in [4].
    Receive: The field MUST be used in accordance with RFC 2327[12].
    11.2. Recvonly
    Send: The field MUST be used in accordance with RFC 2543[11].
    Receive: The field MUST be used in accordance with RFC 2543[11].
    11.3. Sendrecv

Send: The field MUST be used in accordance with RFC 2543[11].
Receive: The field MUST be used in accordance with RFC 2543[11].

11.4. Sendonly

Send: The field MUST be used in accordance with RFC 2543[11], except that the IP address and port number MUST NOT be zeroed.
Receive: The field MUST be used in accordance with RFC 2543[11].

11.5. X-pc-codecs

Send: The field contains a list of alternative codecs that the endpoint is capable of using for this connection. The list is ordered by decreasing degree of preference, i.e. the most preferred alternative codec is the first one in the list. A codec is coded similiarly to <payload type> in an rtpmap.
Receive: Conveys a list of codecs that the remote endpoint is capable of using for this connection. The codecs MUST NOT be used until signaled through a media (m=) line.

11.6. X-pc-Csuites-rtp

Send: The field contains a list of ciphersuites that the endpoint is capable of using for the bearer channel stream of this connection. The first ciphersuite listed is what the endpoint is currently expecting to use. Any remaining ciphersuites in the list represent alternatives ordered by decreasing degree of preference, i.e. the most preferred alternative ciphersuite is the second one in the list. A ciphersuite is encoded as a string of four hexadecimal characters, where the first two characters identify the authentication algorithm and the last two characters identify the encryption algorithm. The actual list of ciphersuites is provided in [2].
Receive: Conveys a list of ciphersuites that the remote endpoint is capable of using for this connection. Any other ciphersuites than the first in the list cannot be used until signaled through a media (m=) line.

11.7. X-pc-Csuites-rtcp

Send: The field contains a list of ciphersuites that the endpoint is capable of using for the RTCP stream of this connection. The first ciphersuite listed is what the endpoint is currently expecting to use. Any remaining ciphersuites in the list represent alternatives ordered by decreasing degree of preference, i.e. the most preferred alternative ciphersuite is the second one in the list. A ciphersuite is encoded as a string of four hexadecimal characters, where the first two characters identify the authentication algorithm and the last two characters identify the encryption algorithm. The actual list of ciphersuites is provided in [2].
Receive: Conveys a list of ciphersuites that the remote endpoint is capable of using for this connection. Any other ciphersuites than the first in the list cannot be used until signaled through a media (m=) line.

11.8. X-pc-secret

Send: The field contains an end-to-end secret to be used for RTP and RTCP security. The secret is encoded similarly to the encryption key (k=) parameter of RFC 2327 [12] with the following constraints: 1) The encryption key MUST NOT contain a ciphersuite, only a passphrase, and 2) the <method> specifying the encoding of the pass-phrase MUST be either “clear” or “base64.”
Receive: The field conveys the end-to-end secret to be used for RTP and RTCP security.

11.9. X-pc-qos

Send: The field MUST be present. The parameter <mandatory> MUST be present and set to “mandatory” in the SDP attached to an INVITE or PRACK, and the parameter value MUST be set to “success” in the SDP attached to a PRECONDITION-MET. The parameter “sendrecv” SHOULD be present. The parameter “confirm” MUST be present in an SDP sent from the destination to the call originator.
Receive: The field conveys the end-to-end qos requirements, and MUST be present.

11.10. X-pc-security

Send: The field MAY be present.
Receive: The field conveys the end-to-end security precondition requirements, and MAY be present.

11.11. Ptime

Send: The ptime SHOULD always be provided and when used it MUST be used in accordance with RFC 2327[12]. When an rtpmap or non well-known codec is used, the ptime MUST be provided.
Receive: The field MUST be used in accordance with RFC 2327[12]. When "ptime" is present, the MTA MUST use the ptime in the calculation of QoS reservations. If "ptime" is not present, the MTA MUST assume reasonable default values for well-known codecs.

12. Media Announcements (m=) consists of 3 sub-fields:
M= <media> <port> <transport> <format>
M= audio 3456 RTP/AVP 0

12.1. Media:
Send: The 'audio' media type MUST be used.
Receive: The type received MUST be 'audio'

12.2. Port
Send: MUST be filled in accordance with RFC2327[12]. The port specified is the receive port, regardless of whether the stream is unidirectional or bi-directional. The sending port may be different.
Receive: MUST be used in accordance with RFC 2327[12]. The port specified is a receive port. The sending port may be different.

12.3. Transport: "RTP/AVP" MUST be used.
Send: The transport protocol 'RTP/AVP' MUST be used.
Receive: The transport type MUST be 'RTP/AVP'.

12.4. Media Formats:
Send: Appropriate media type as defined in RFC 2327[12] MUST be used.
Receive: In accordance with RFC 2327[12].

6. MTA interfaces (MTA to CMS/Proxy and MTA-MTA)

This section discusses the call signaling requirements for an MTA that uses the DCS model. An MTA is an untrusted SIP UAC/UAS, as discussed in section 2.4. Each client (MTA) in the network generally communicates with a single CMS/Proxy. Each CMS/Proxy may communicate with many MTAs and (in general) a much smaller number of CMTSs. The name or address of the CMS/Proxy with which a particular MTA communicates is either provisioned or learned (see PacketCable OSS specification [6]).

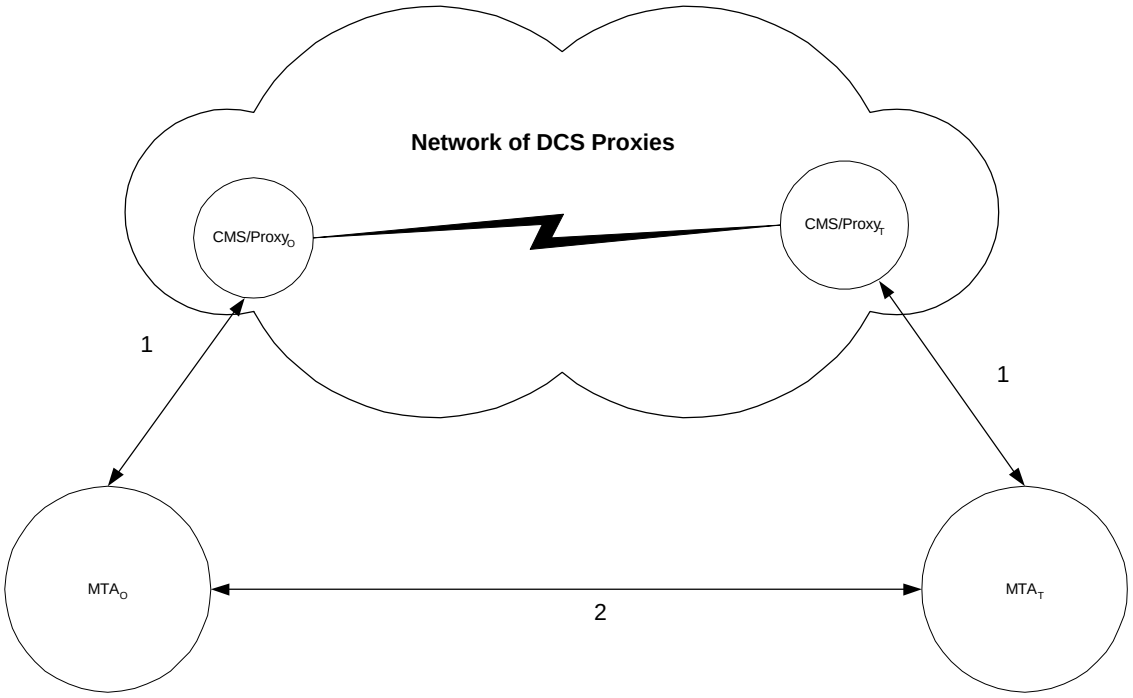


Figure 7: Paths for signaling messages

This section describes the messages required to support IP Telephony between MTAs that use Distributed Call Signaling. The messages sent from one MTA to the other may be either sent directly or through a network of CMS/Proxies.

The discussion in this section, and the diagram above, show only interfaces with other MTAs supported by CMS/Proxies. The CMS/Proxy is one type of CMS, as described in section 7, that can interoperate with all other CMSs. Therefore, the message set and message procedures defined here enable communication with a Residential Gateway (RGW) controlled by a Call Agent implementing Network-based Call Signaling (NCS), and enable communication with a Trunk Gateway (TGW) controlled by a Media Gateway Controller implementing the Trunk Gateway Control Protocol (TGCP). See Section 7 for a discussion of the types of CMSs and their interface requirements. For simplicity in the current section, all the examples and discussion are based on connections between MTAs supported by CMS/Proxies.

In Figure 7, MTA_O refers to the MTA where the call originates and MTA_T refers to the MTA that receives the call (where the call terminates) in a two-party phone call. The CMS/Proxy associated with the MTA originating a call is referred to as the originating CMS/Proxy and is denoted $CMS/Proxy_O$. The CMS/Proxy

associated with MTA_T is referred to as the terminating CMS/Proxy and is denoted by $CMS/Proxy_T$. Messages for setting up a new call or changing call parameters/call legs of an active call go through the CMS/proxies. In the above figure, path labeled (1) shows the path taken by the signaling messages from one MTA to the other through CMS/proxies. Direct signaling between MTAs follows the path labeled (2).

6.1 SIP Message Definition Overview

An INVITE message and 183-Session-Progress response is used to exchange capabilities and set-up call state in the network prior to alerting the user. The INVITE message and its status response go through the CMS/Proxies. Successful completion of the initial handshake triggers the resource reservation process. Following a successful end-to-end reservation of resources, a PRECONDITION-MET message is sent directly to the callee, which triggers a 180-Ringing response and (when the call is answered) a 200-OK final response. Each of the provisional responses to the INVITE request generate a PRACK/200-OK exchange to confirm the delivery of the provisional response.

The DCS architecture extends the use of the basic INVITE/response/ACK transaction. Variants of the SIP INVITE method for the DCS architecture are: INVITE(also,replace), INVITE(hold), INVITE(resume), INVITE(return-call), INVITE(call-trace), INVITE(also), INVITE(BLV) and INVITE(EI).

INVITE(replace) and INVITE(also) messages change the call description by creating or tearing down call legs. These messages are sent through CMS/Proxies so that appropriate reservation and billing modifications are made and transferred to network entities (participating end-points, the CMTS and record keeping servers).

INVITE(hold) and INVITE(resume) change the SDP description on an active call. These messages use direct signaling between the MTAs as they do not require change in billing or reservation for the call.

INVITE(return-call) and INVITE(call-trace) are initiated by the destination of a previous call, and require tracing the path back to the caller for callback feature or reporting to law enforcement, respectively. Both these messages go through the CMS/Proxies.

Operator services of Busy line verification and Emergency interrupt use the INVITE(BLV) and INVITE(EI) messages from the PSTN media gateway to the MTA being verified/interrupted. BLV will be signaled through CMS/Proxies, while EI will be directly signaled. These messages have an extension header (Dcs-OSPS) that causes the receiving MTA to not return busy but instead make copies of the media stream (for busy line verification) or switch to the incoming call (for emergency interrupt).

Consider a call with the following properties:

Attributes associated with end-points	Origination	Destination	Example
Name	User-o	User-t	John Doe
Hostname	Host-o	Host-t	mta.com
MTA address	Host(mta-o)	Host(mta-t)	44.20.0.3, or mta.provider
MTA port number	Port(mta-o)	Port(mta-t)	1234
CMS address	Host(dp-o)	Host(dp-t)	192.136.26.6, or dp.provider
Telephone number	E.164-o	E.164-t	123-456-7890
CMTS address	Host(CMTS-o)	Host(CMTS-t)	20.20.10.8, or cmts-o.provider
CMTS port number	Port(CMTS-o)	Port(CMTS-t)	4321

Gate ID at CMTS	GID-o	GID-t	

Attribute associated with Calls	**Notation**	**Comments**
Call-ID	ID	Random string, unique within a call. Suggested implementation is a base64 encoding of a SHA-1 or MD5 cryptographic hash of local provisioned parameters (e.g. phone number) combined with a timestamp and a sequence number.
Call Sequence Number	n_o	Random starting sequence number chosen by MTA_O for the initial INVITE request.
	n_o+1 n_o+2 n_o+3 etc.	Numeric value one (or two, or three) greater than the Call-Sequence-Number value used in initial INVITE request sent by MTA_O for the same call leg.
	n_i	Call sequence number value used in a mid-call request message. If sent by MTA_O, this value is one greater than the most recent request sent by MTA_O for this call leg. Each client has an independently managed call sequence number for each call instance at that client. If the first request sent by MTA_T, this value is a random starting sequence number chosen by MTA_T (n_t). If a subsequent request sent by MTA_T, this value is one greater than the most recent request sent by MTA_T for this call leg.
Provisional Response Sequence Number	x_t x_t+1 x_t+2 etc.	Sequence number used in requesting an acknowledgement to a provisional response. If the first request for PRACK, this value is a random starting sequence number. If a subsequent request for a PRACK, this value is one greater than the most recent provisional response sequence number sent.

All signaling messages are based on the Session Initiation Protocol (SIP), as specified in RFC 2543[11]. Necessary extensions and changes to the protocol specified in the RFC are presented in Section 3. The MTA MUST support the INVITE, ACK, BYE, CANCEL, PRACK, PRECONDITION-MET and REGISTER request methods, and MAY support the OPTIONS Request method. The MTA MUST be capable of generating status responses to all valid SIP requests.

6.2 MTA Retransmission, Reliability, and Recovery Strategy

The MTA MUST implement a retransmission timer to recover from lost request message. SIP [11] defines a scheme based on two timer values, T1 and T2, where the retransmission interval starts at T1 seconds, and

is doubled, with each attempt (up to a limit of T2 seconds), with a maximum number of retransmissions. DCS compliant MTAs MUST, at a minimum, allow the value of T1 to be dependent on the request message being sent, and SHOULD implement a retransmission strategy using exponential back-off, and configurable initial and maximum retransmission timer values.

In addition to the mechanisms defined in [11], DCS compliant MTAs MUST implement an additional timer, called T3 in this specification, that starts at certain predetermined events in the call setup sequence. On expiration of this timer, the MTA MUST abort the current request and return to a known idle state. On receipt of the first provisional response to an INVITE, the originating MTA sets this timer to value T-setup. On receipt of a 180-Ringing provisional response to an INVITE, the originating MTA resets this timer to T-Ringback. On receipt of a final response, the originating MTA cancels this timer. On receipt of an INVITE message, the terminating MTA sets this timer to T-Resource. On sending 180-Ringing, the terminating MTA resets this timer to T-Ringing. On receipt of ACK, the terminating MTA cancels this timer. Default values for all of these timers (T-setup, T-Ringback, T-Resource, and T-Ringing) are given in Appendix A.

When the provisioned number of message retransmissions is exceeded for an INVITE without any responses, the MTA MUST try a different proxy address, if available. If the list of proxy addresses was obtained by a DNS lookup, and it reaches end of the proxy address list, the MTA SHOULD perform a new DNS lookup of the FQDN of its proxy. When a provisioned number (which may be infinite) of proxies have been tried, the MTA MUST abort the current request and return to a known idle state.

An MTA receiving a SIP request MAY send a 100-Trying provisional response to any request, and SHOULD send the 100-Trying provisional response if another (provisional or final) response will not be sent within 200ms of receipt of the request.

The MTA SHOULD use the reliable-provisional-response facility of section 3.2.1 to ensure delivery of all provisional responses other than the 100-Trying.

6.3 General Requirements for headers

The table below lists general syntax and processing requirements for SIP header extensions in SIP messages received or sent by MTAs. The table also lists any additional requirements or exceptions for standard headers in SIP messages received or sent by MTAs. All other headers are processed by the MTA according to the requirements listed in RFC2543[11].

Header Name	*Direction*	*Presence*	*Requirements, Comments*
Request Line	*MTA to CMS/Proxy*	*!*	*MUST conform to rules for DCS-URLs as stated in section 4.2* *If private-param is present, CMS/Proxy decrypts and validates the DCS-URL contents using its private key.*
	CMS/Proxy to MTA	*!*	*MUST conform to rules for DCS-URLs as stated in section 4.2, but MUST NOT have private-param in request line.* *MUST identify a line termination on the MTA in the initial INVITE request of the call leg.*
Via	*MTA to CMS/Proxy*	*!*	*MUST be IP address or FQDN of MTA* *In responses from MTAs: CMS/Proxy MUST recover and replace the unencrypted Via list saved from request, and MUST forward the response to the address in the next Via header.*
	CMS/Proxy to MTA	*!*	*CMS/Proxy copies and saves the Via headers received in all requests sent to MTAs.* *The topmost Via header is added by the CMS/Proxy, is an IP address or FQDN, and is unencrypted.*
Supported	*MTA to CMS/Proxy*	*!*	*MUST be present, and include "org.ietf.100rel"*
	CMS/Proxy to MTA	*!*	*Is forwarded without modification*
From	*MTA to CMS/Proxy*	*!*	*MUST be provided by MTA_O to $CMS/Proxy_O$ in initial INVITE*
	CMS/Proxy to MTA		*Is forwarded without modification*

To	*MTA to CMS/Proxy*	*!*	*MUST be provided by MTA_O to CMS/$Proxy_O$ in initial INVITE*
	CMS/Proxy to MTA		*Is forwarded without modification*
Call-ID	*MTA to CMS/Proxy* *CMS/Proxy to MTA*	*!*	*MUST be provided by MTA_O to CMS/$Proxy_O$ in initial INVITE*
			Is forwarded without modification
Contact	*MTA to CMS/Proxy*	*O*	*MUST be provided by MTA_O to CMS/$Proxy_O$ in initial INVITE. MUST be provided by MTA_T in first 1xx, 2xx, or 3xx (except 100) response of call-leg to CMS/$Proxy_T$.* *In an INVITE or 1xx-2xx response, MUST be a SIP-URL in either IPv4 or FQDN form, as described in section 4.2, and MUST NOT contain a private-param* *In a 3xx response, MUST be present, and be either a SIP-URL (possibly with "user=phone") or Tel: URL*
	CMS/Proxy to MTA	*O*	*Is provided by CMS/$Proxy_T$ to MTA_T in initial INVITE. Is provided by CMS/$Proxy_O$ in first 1xx, 2xx, or 3xx (except 100) response of call-leg to MTA_O.* *In an INVITE or 1xx-2xx response, is a SIP-URL in either IPv4 or FQDN form, as described in section 4.2, and does not contain a private-param* *In a 3xx response, is a SIP-URL containing a private-param*
Dcs-Media-Authorization	*MTA to CMS/Proxy*	*X*	*MUST NOT be present in messages sent by MTAs to CMS/Proxies.*
	CMS/Proxy to MTA	*O*	*Is provided by CMS/$Proxy_T$ to MTA_T in initial INVITE of call-leg.* *Is provided by CMS/$Proxy_O$ in first 183 response of call-leg to MTA_O.*
Dcs-State	*MTA to CMS/Proxy*	*O*	*MTA MUST include all previous Dcs-State headers with matching call-leg identifiers in messages from MTA to CMS/Proxy.*
	CMS/Proxy to MTA	*O*	*Is provided by CMS/$Proxy_T$ to MTA_T in initial INVITE.* *Is provided by CMS/$Proxy_O$ in first 183 response of call-leg to MTA_O.* *Dcs-State is encrypted with the private key of a CMS/Proxy and only has significance to the Proxy that generates the data.*
Dcs-Remote-Party-ID	*MTA to CMS/Proxy*	*O*	*SHOULD be provided by MTA_O to CMS/$Proxy_O$ in initial INVITE, and verified by CMS/$Proxy_O$.* *SHOULD be present in INVITE requests that contain Also or Replaces headers.* *SHOULD be provided by MTA_T in first non-100 response to CMS/$Proxy_T$, and verified by CMS/$Proxy_T$*
	CMS/Proxy to MTA	*O*	*Is provided by CMS/$Proxy_T$ to MTA_T in INVITE message.* *Is provided by CMS/$Proxy_O$ to MTA_O in first 1xx, 2xx, or 3xx (except 100) response to INVITE.*
Dcs-Anonymity	*MTA to CMS/Proxy*	*O*	*MAY be provided by MTA_O to CMS/$Proxy_O$ in initial INVITE.* *MAY be provided by MTA_T in first non-100 response of call-leg to CMS/$Proxy_T$.* *MAY be present in INVITE requests that contain Also or Replaces headers.* *MUST be provided if MTA desires anonymity. If not present, default value is "Off"*
	CMS/Proxy to MTA	*X*	
Dcs-Also	*MTA to CMS/Proxy*	*O*	*MAY be sent by MTA in INVITE requests.* *MUST contain a DCS-URL.* *MAY include private-param in the DCS-URL.*
	CMS/Proxy to MTA	*O*	*Is sent by CMS/Proxy in INVITE requests.* *Contains a DCS-URL.with a private-param encrypted with CMS/Proxy's private key before forwarding to MTA.*
Dcs-Replaces	*MTA to CMS/Proxy*	*O*	*MAY be sent by MTA in INVITE requests.* *MUST be a SIP-URL.* *MAY be appended to Dcs-Also header in proxied INVITE requests.*
	CMS/Proxy to MTA	*O*	*Is sent by CMS/Proxy in INVITE requests.*
Dcs-Billing-Info	*MTA to CMS/Proxy*	*X*	*MUST NOT be present in messages sent by MTAs to CMS/Proxies.*
	CMS/Proxy to MTA	*X*	
Dcs-Billing-ID	*MTA to CMS/Proxy*	*X*	*MUST NOT be present in messages sent by MTAs to CMS/Proxies.*
	CMS/Proxy to MTA	*X*	
Dcs-OSPS	*MTA to CMS/Proxy*	*O*	*MUST be forwarded unchanged in all proxied requests.*
	CMS/Proxy to MTA	*O*	*Is forwarded unchanged in all proxied requests.*
Dcs-LAES	*MTA to CMS/Proxy*	*X*	*MUST NOT be present in messages sent by MTAs to CMS/Proxies*
	CMS/Proxy to MTA	*X*	
Dcs-Redirect	*MTA to CMS/Proxy*	*X*	*MUST NOT be present in messages sent by MTAs to CMS/Proxies*
	CMS/Proxy to MTA	*X*	

KEY:

Code:	Meaning	Direction:	Requirements
!	MUST be present	MTA to CMS/Proxy	MTA MUST supply this header. CMS/Proxy MUST verify its contents
		CMS/Proxy to MTA	CMS/Proxy MUST supply this header. MTA MUST verify its contents
X	MUST NOT be present	MTA to CMS/Proxy	MTA MUST NOT supply this header CMS/Proxy MUST verify it is not present.
		CMS/Proxy to MTA	CMS/Proxy MUST NOT supply this header MTAs MAY verify it is not present
O	MAY be present		No requirement beyond those specified in table above

6.4 SIP Messages for Basic Call Setup

The basic INVITE message sequence for a DCS call setup include the INVITE/183-Session-Progress/180-Ringing(optional)/200-OK/ACK exchange, a PRECONDITION-MET/200-OK exchange, and one or two PRACK/200-OK message exchanges. These are discussed in the following subsections.

The initial INVITE message and the status responses to the INVITE go through the CMS/Proxies. The PRACK messages, the PRECONDITION-MET message, and the response to the INVITE's final status response (typically the ACK request) is directly sent end-to-end between the caller and the callee.

The following sections trace a basic call from origination to completion, and give the requirements for each message exchange. It therefore switches viewpoints, from origination to termination, and back. For procedures followed by MTA_O initiating a call, see sections 6.4.1, 6.4.3, 6.4.6, and 6.4.8. For procedures followed by MTA_T in terminating a call, see sections 6.4.2, 6.4.4, 6.4.5, and 6.4.7. A conformant MTA MUST implement the procedures in all of these subsections.

The architecture extends the syntax of the INVITE message SDP body with an attribute that permits caller and callee to exchange capabilities and to reserve necessary network resources prior to alerting the user. The initial exchange consists of INVITE followed by 183-Session-Progress and a provisional acknowledgement (PRACK). Following this exchange, both MTAs know sufficient information to reserve the resources that will be needed to complete the call. Once those resources have been reserved, the call originator sends a PRECONDITION-MET message, and the destination continues the normal SIP processing with a 180-Ringing or 200-OK.

The behavior below also shows the procedures for call forwarding (unconditional and busy) and call forwarding (no answer).

6.4.1 MTA_O Sending INVITE to $CMS/Proxy_O$ initiating a call

In order to initiate a connection, an MTA MUST send a SIP INVITE message to its CMS/Proxy. The format of the INVITE message sent by the MTA and the requirements on the header fields are as follows.

INVITE: (MTA_O -> CMS/Proxy) Header:	***Requirements on MTA for message generation and on $CMS/Proxy_O$ for message checking***
INVITE DCS-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST conform to the rules for DCS-URLs as given in 4.2.* *The hostname MUST be $CMS/Proxy_O$.*

Via: SIP/2.0/UDP Host(mta-o)	*MUST be present.* *MUST contain the IP address or FQDN of the originating MTA.* *MUST represent the same calling party as Contact: and Dcs-Remote-Party-ID: headers*
Supported: org.ietf.sip.100rel	*MUST be present. MUST indicate org.ietf.sip.100rel*
Dcs-Remote-Party-ID: [USER-o] <tel:E.164-o>	*SHOULD be present.* *Represents the same calling party as Via: and Contact: headers.* *URL MUST contain the phone number of the calling party, either as a tel: URL, or as a SIP-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the calling party.*
Dcs-Anonymity:	*MUST be present if caller desires anonymity. Otherwise MAY be present.* *MUST be either OFF, FULL, URL, NAME, IPADDR, or a valid combination as given in 3.3.3.* *If the caller has not requested privacy, MUST set to OFF.* *If the caller has requested privacy, MUST set to FULL, or a combination of URL, NAME, and IPADDR.*
From:	*The From: header MUST be present, and MUST follow the requirements of section 4.6.6.* *It MAY identify the caller by name, IP address, or by phone number* *If Dcs-Anonymity is FULL, URL, Name or IPAddr, the username in addr-spec MUST be a random string that ensures privacy of the caller, the hostname MUST be the non-identifying name "localhost", and the display-name MUST be omitted.* *The triple (From, To, CallID) uniquely identifies the call at MTA_O and MTA_T*
To:	*The To: Header MUST be present, and MUST follow the requirements of section 4.6.7.* *The To: header MAY contain the URI of the callee* *If Dcs-Anonymity is OFF, then the To: header SHOULD contain a tel: URI with the dialed digits.* *If Dcs-Anonymity is FULL or URL or NAME, then the username in addr-spec MUST be a random string that is different from the From: header, the hostname MUST be the non-identifying name " localhost", and the display-name MUST be omitted.*
Call-ID: ID	*MUST be present.* *MUST be a unique string. Call-ID is an ASCII encoding of a random number designed to be unique over a period of several months.*
CSeq: n_o INVITE	*MUST be present.* *Call sequence number "n_o" and the request method MUST be present.*
Contact: sip:Host(mta-o)	*MUST be present. MUST be a SIP-URL in either IPv4 or FQDN form.* *MUST represent the same endpoint as the Via: header*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The session description MUST include the list of CODECs MTA_O is willing to support for this connection. It MUST include on the media (m=) line the preferred CODEC for which resources MUST be available in the MTA, and available to receive and play payload packets. It MAY include an attribute (a=X-pc-codecs) line giving alternatives available.* *The SDP MUST include a qos precondition of the form "a=X-pc-*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A.

If any of the required fields are missing or if any of the required fields are improperly formatted, the CMS/Proxy (CMS/Proxy$_O$) MUST respond with an appropriate 4xx, 5xx, or 6xx error code. CMS/Proxy$_O$ SHOULD record this error event to the syslog server.

CMS/Proxy$_O$ locates the CMS/Proxy associated with the address in the Request-URI of the received message. It adds the Dcs-Gate, Dcs-Billing-Info and Dcs-Billing-ID headers to the INVITE request and sends it to the CMS/Proxy associated with the terminating MTA.

CMS/Proxy$_O$ MAY send a 100-Trying provisional response to MTA$_O$, and MUST send the 100-Trying provisional response if it is unable to generate a provisional or final response within 200ms. The 100-Trying message MUST be as described in the following table.

100-Trying: (CMS/Proxy$_O$ -> MTA$_O$) Header :	***Requirement of CMS/Proxy for message generation Requirement of MTA for message checking***
SIP/2.0 100 Trying	*Status line with status code 100 MUST be present.*
Via: SIP/2.0/UDP Host(mta-o)	*MUST be copied from received INVITE message.*
From:	*From, To, CallID and CSeq MUST be copied from received INVITE message, and a tag-param MAY be added, as per RFC2543 [11].*
To:	
Call-ID:	
CSeq:	

On receipt of a 100-Trying provisional response, the retransmission timer T1 MUST be cancelled, and the transaction timer (T3) for this exchange SHOULD be set to T-setup. The default value of (T-setup) is given in Appendix A. On expiration of T3, the MTA MUST clear the call attempt, and send a CANCEL message to its CMS/Proxy with the same values of Request-URI, From, To, and CallID, and any Dcs-State headers received for this call attempt.

6.4.2 MTA$_T$ receives Invite from CMS/Proxy$_T$

The terminating CMS/Proxy (CMS/Proxy$_T$) receives an INVITE message and locates the IP address of the called MTA. CMS/Proxy$_T$ removes the Dcs-Billing-Info and Dcs-Billing-ID headers from the received message. These headers are not present in the INVITE message sent to the MTA. It formulates the Dcs-State information and encrypts it. The Dcs-State header is inserted in the INVITE sent to the MTA.

CMS/Proxy$_T$ inserts the Dcs-Media-Authorization header. This header contains the Gate identification information for this particular call in the terminating access network.

If the caller has requested privacy with Dcs-Anonymity: Full or URL, CMS/Proxy$_T$ replaces the URL in the Dcs-Remote-Party-ID header with a private URL and adds “rpi-id=private.” The private URL is formed by encrypting the original URL with CMS/Proxy$_T$’s privately held key, placing the resulting string as the username, inserting the proxy name as hostname, and adding a url-parameter of “private.” If the caller has requested privacy with Dcs-Anonymity: Full or Name, CMS/Proxy$_T$ deletes the display-name in the Dcs-Remote-Party-ID header. If the callee has not subscribed to calling-name delivery, then CMS/Proxy$_T$ deletes the display-name in the Dcs-Remote-Party-ID header. If the callee has not subscribed to calling-number delivery, then CMS/Proxy$_T$ replaces the URL in the Dcs-Remote-Party-ID header with a private URL (as described above) and adds “rpi-id=na.” CMS/Proxy$_T$ deletes the Dcs-Anonymity header from the INVITE message.

CMS/Proxy$_T$ inserts its address in the topmost via. All other via’s received in the message from the originator are hidden from MTA$_T$. CMS/Proxy$_T$ encrypts these via’s in the INVITE message sent to MTA$_T$, and includes them either in the Via header or as part of the Dcs-State information.

CMS/Proxy$_T$ forwards the resulting INVITE to MTA$_T$. The request that is sent to MTA$_T$ MUST adhere to the requirements given in the table below.

Invite: ($CMS/Proxy_T$ -> MTA_T) Header:	***Requirement of CMS/Proxy***
INVITE DCS-URL SIP/2.0	*MUST be present.* *MUST be sufficient for the MTA to determine the proper line being addressed.*
Via: SIP/2.0/UDP Host(dp-t);branch=x, a	*At least a single Via: header MUST be present. The topmost Via MUST be unencrypted and MUST contain the IP address or FQDN of the terminating CMS/Proxy. MUST include branch=x, where x is a unique value for this transaction. Other VIA headers MAY be encrypted. If Dcs-Anonymity is FULL or IPADDR, then the Via headers MUST be encrypted.*
Supported: org.ietf.sip.100rel	*MUST be present. MUST indicate org.ietf.sip.100rel*
Dcs-Remote-Party-ID: [display-name] <DCS-URL>	*MUST be present. MUST be modified per the requirements of section 7.6.2.*
Dcs-Media-Authorization: GID-t	*MUST be added by $CMS/Proxy_T$. MUST contain the Gate-ID to be used for resource reservation at MTA_T.*
Dcs-State: Host(dp-t);{DS-t}$_K$;	*MUST be present.* *MUST be added by $CMS/Proxy_T$, and contain call-information as needed by $CMS/Proxy_T$ for proper handling of call features.* *MUST be encrypted with $CMS/Proxy_T$'s private key.*
From:	*MUST be present.* *MUST be copies of same headers received in the request from $CMS/Proxy_O$.*
To:	
Call-ID:	
CSeq:	
Contact: sip: Host(MTA-o | Ann-t)	*MUST be present.* *If Dcs-Anonymity: header is set to Full or IPADDR, Contact: header MUST contain IP address of an Anonymizer ; otherwise MUST be a copy of the Contact: header received in the request from $CMS/Proxy_O$.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
V= o= s= m= c= a=X-pc-secret:	*MUST be present.* *MUST contain a line with "a=X-pc-secret" providing the encryption key (e.g. Clear:RC4/WhenInTheCourseOfHumanEvents)* *MUST contain a qos precondition of the form "a=X-pc-qos:mandatory"*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T- proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of any response.

When the provisioned number of message retransmissions is exceeded for an INVITE without any responses, the proxy MUST try a different MTA address, if available. When all of the MTA addresses have been tried (subject to a provisionable maximum number), the proxy MUST consider the MTA unreachable. The CMS/Proxy MUST return a 480-Temporarily-Unavailable error response, or, if the MTA has subscribed and registered for call-forwarding, return a 302-Moved-Temporarily (described in section 7.6.2.3).

An MTA MUST be capable of receiving an INVITE message from its CMS/Proxy at any time. The INVITE message received at the terminating MTA is shown in the table below. The Requirements shown for this message specify the actions required of the MTA if the field is not present or is not in the correct format.

An MTA receiving an INVITE MUST use information in the Dcs-Remote-Party-ID header for calling-identity delivery, which is the information verified by the Proxies. Information contained in the display-

name string in the From header MUST NOT be used as authenticated calling-identity, as this is supplied by the originating user and not verified by the Proxies.

Invite: ($CMS/Proxy_T$ -> MTA_T) Header:	***Requirement of MTA***
INVITE sip:E.164-t | user-t @Host(mta-t); user=phone | ip SIP/2.0	*MUST be present. It MUST be sufficient for the MTA to determine the proper line being addressed.*
Via: SIP/2.0/UDP Host(dp-t), a	*At least a single Via: header MUST be present.*
Supported: org.ietf.sip.100rel	*MUST be present. MUST indicate org.ietf.sip.100rel*
Dcs-Remote-Party-ID: [display-name] <DCS-URL>	*MUST be present, and as described in Section 3.3.1. MUST be used for Calling-Number and Calling-Name delivery service.*
Dcs-Media-Authorization: GID-t	*MUST be present. Value MUST be used for resource reservations..*
Dcs-State:	*MUST be present.*
From:	*From:,To:, Call-ID:, and CSeq MUST be present. These are a direct copy of the corresponding headers from the INVITE message sent by originating MTA through its CMS/Proxy. The terminating CMS/Proxy does not modify any of these header fields.*
To:	
Call-ID:	
CSeq:	
Contact:	*MUST be present..*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (...)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*SDP description of media stream requested by the call originator MTA_O.*

If the headers indicated in the above table are not present in the received INVITE, the MTA MUST not accept the call and MUST generate the appropriate client 4xx response, as shown in 6.4.2.3.

The MTA stores the INVITE message for the duration of the call. The Dcs-State header values received in the INVITE MUST be included in all requests and responses sent to the proxy with the matching call-leg identification (From, To, and Call-ID).

The Dcs-Media-Authorization header value is used for resource allocation.

The value of Dcs-Remote-Party-ID is used for mid-call changes, such as transfer and three-way-calling.

6.4.2.1 MTA_T sending 183-Session-Progress Status response to INVITE

MTA_T examines the capability parameters in the SDP part of the message (the m= line) and determine which media channel parameters it can accommodate for this call.

If the MTA is willing to accept the call, a response as shown below MUST be sent to the address in the topmost Via header (typically MTA_T's proxy).

MTA_T MUST send a 183-Session-Progress response including the following headers and contents. The response's session description MUST indicate the CODECs that MTA_T is willing to support, and MUST be a subset of those received in the INVITE. Upon receiving the 183-Session-Progress message, $CMS/Proxy_T$ MUST verify that all required headers and fields are present and formatted as shown in the table below.

183-Session-Progress: (MTA_T -> $CMS/Proxy_T$) Header :	***Requirement of MTA for message generation Requirement of CMS/Proxy for message checking***

SIP/2.0 183 Session Progress	*Status line with status code 183 MUST be present.*
Via: SIP/2.0/UDP Host(dp-t)	*MUST be copied from received INVITE message. If other Via's in encrypted form are present, MTA_T MUST copy them in this response.*
Dcs-Remote-Party-ID: [User-t] <tel:E.164-t>	*SHOULD be present.* *Represents the same calling party as d Contact: header.* *URL MUST contain the phone number of the called party, either as a tel: URL, or as a DCS-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the called party.*
Dcs-Anonymity:	*If the Call-ee has requested privacy, this header MUST be present and MUST be FULL, URL, NAME, or IPADDR. If the callee has not requested privacy, this header MAY be present, and if present, MUST be OFF.*
Dcs-State:	*MUST be copied from the INVITE message*
From:	*From, To, CallID and CSeq MUST be copied from received INVITE message, and a tag-param MAY be added, as per RFC2543 [11].*
To:	
Call-ID:	
CSeq:	
Contact: sip:Host(mta-t)	*MUST be inserted by MTA_T as the address for future direct signaling messages to MTA_T. MUST be a SIP-URL in either IPv4 or FQDN form. MUST represent the same party as Dcs-Remote-Party-ID.*
Session: qos	*MUST be present. MUST containe 'qos' and MAY also contain 'security'. MUST NOT contain 'media'*
Rseq: x_t	*MUST be present. MUST contain the initial random sequence number chosen by MTA_T.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..* *The response to INVITE must contain the SDP description of the media stream to be sent to MTA_T.*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
V= o= s= c= b= t= a= m=	*MUST be present.* *SDP description of media streams acceptable to MTA_T, as described in Section 5.* *MUST contain a line 'a=X-pc-qos:mandatory' with attribute 'confirm'*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of the matching PRACK.

6.4.2.2 MTA_T sending 3xx-Redirect Status response to INVITE

If MTA_T wishes to forward the call (e.g. if call-forwarding-unconditional or call-forwarding-busy is enabled at the MTA), a 302-Redirect status response MUST be sent to $CMS/Proxy_T$ with the forwarded-to destination URI in the contact header.

302-Redirect: (MTA_T -> $CMS/Proxy_T$) Header	***Requirement on MTA for message generation Requirement on CMS/Proxy for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST be inserted by MTA_T. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*MUST be copied from the INVITE message*

Dcs-Remote-Party-ID: [User-t] <tel:E.164-t>	*SHOULD be present.* *Representa the same calling party as Contact: header.* *URL MUST contain the phone number of the called party, either as a tel: URL, or as a DCS-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the called party.*
Dcs-Anonymity: OFF | FULL | URL | NAME | IPADDR	*If the Call-ee has requested privacy, this header MUST be present and MUST be FULL, URL, NAME, or IPADDR. If the callee has not requested privacy, this header MAY be present, and if present, MUST be OFF.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be inserted by MTA_T and carries the new destination information. It MUST be a valid URI.* *If the new destination is a telephone number, then the format of the URI MUST be a tel: URI where the URI contains the sequence of dialed digits, including any prefixes.*
Expires:	*MAY be present*

The originating MTA's CMS/Proxy processes the redirect 3xx response.

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

If MTA_T does not subscribe to the Call Forwarding service, or if any of the header verification checks fail, $CMS/Proxy_T$ MUST send a 480-Temporarily-Unavailable error response to $CMS/Proxy_O$, and MUST send a CANCEL to MTA_T. Otherwise, $CMS/Proxy_T$ MUST send an ACK message to MTA_T. The required fields of the message are as shown below. The transaction between MTA_T and $CMS/Proxy_T$ is now complete.

ACK: ($CMS/Proxy_T$ -> MTA_T) **Header:**	***Requirement at CMS/Proxy***
ACK DCS-URL SIP/2.0	*The Response line MUST be present.*
Via:	*MUST be present. MUST be the IP address or FQDN of $CMS/Proxy_T$.*
From:	*MUST be present.* *MUST be copies of same headers in Request from $CMS/Proxy_O$.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in request from $CMS/Proxy_O$, method MUST indicate ACK*

6.4.2.3 MTA_T sending Other Status Response to INVITE request

A final error response, 4xx, 5xx, or 6xx response, MUST be sent as per [11]. This includes, but is not limited to, 486-Busy Here. The error response MUST be generated as follows.

Error: (MTA_T -> $CMS/Proxy_T$) **Header:**	***Requirement on MTA for message generation*** ***Requirement on CMS/Proxy for message checking***
SIP/2.0 xxx	*Status line header MUST be inserted by MTA_T. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*

Dcs-State:	*MUST be copied from the INVITE message*
Dcs-Remote-Party-ID: [User-t] <tel:E.164-t>	*SHOULD be present.* *Represents the same calling party as Contact: header.* *URL MUST contain the phone number of the called party, either as a tel: URL, or as a DCS-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the called party.*
Dcs-Anonymity:	*If the Call-ee has requested privacy, this header MUST be present and MUST be FULL, URL, NAME, or IPADDR. If the callee has not requested privacy, this header MAY be present , and if present, MUST be OFF.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

CMS/Proxy$_T$ MUST send an ACK message to acknowledge the error response.

ACK: (CMS/Proxy$_T$ -> MTA$_T$) **Header:**	***Requirement at CMS/Proxy for message generation*** ***Requirement at MTA for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS/Proxy$_T$.*
From:	*MUST be present.* *MUST be copies of same headers in Request from CMS/Proxy$_O$.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in request from CMS/Proxy$_O$, method MUST indicate ACK*

6.4.3 MTA$_O$ receives 183-Session-Progress Status from CMS/Proxy$_O$

The 183-Session-Progress status message sent by CMS/Proxy$_O$ to MTA$_O$, and the requirements on the headers, is shown in the following table:

183-Session-Progress: (CMS/Proxy$_O$ -> MTA$_O$) **Header:**	***Requirement for message generation at CMS/Proxy$_O$*** ***Requirement for message checking at MTA$_O$***
SIP/2.0 183 Session Progress	*Status line MUST be present, and MUST contain 183*
Via: SIP/2.0/UDP Host(mta-o)	*MUST be present. MUST be copy of Via: header from initial INVITE.*
Dcs-Media-Authorization: GID-o	*MUST be inserted by CMS/Proxy$_O$.* *MUST be present in the message received at MTA$_O$.*
Dcs-State: Host(dp-o); {DS-o}$_K$	*MUST be inserted by CMS/Proxy$_O$, containing call information needed by CMS/Proxy$_O$ in handling mid-call services, and formatted as an ASCII encoding of a structure encrypted by CMS/Proxy$_O$ with a privately-held key.* *MUST be present in the message received at MTA$_O$.*
Dcs-Remote-Party-ID	*MUST be present*
From:	*From, To, Call-ID and Cseq headers MUST be present.* *MUST match an existing call leg.*
To:	
Call-ID:	
Cseq:	

Contact:	*MUST be present. It represents the address for future direct signaling messages to MTA_T.* *MUST be a copy of the Contact: header received in the response from $CMS/Proxy_T$, or anonymization thereof.*
Session: qos	*MUST be present. MUST containe 'qos' and MAY also contain 'security'. MUST NOT contain 'media'*
Rseq: x	*MUST be present. MUST contain a sequence number.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present, and contains the SDP description as given by the terminating MTA The originating CMS/Proxy uses the list of codecs specified in the SDP to authorize maximum resources that may be used during this call at the originating CMTS.* *MUST contain a line 'a=X-pc-qos:' with attribute 'confirm'*

If the 183-Session-Progress provisional response was the first response to the sent INVITE, the retransmission timer T1 MUST be cancelled, and the transaction timer (T3) for this exchange SHOULD be set to T-setup. The default value of (T-setup) is given in Appendix A. On expiration of T3, MTA_O MUST clear the call attempt, and send a CANCEL message to $CMS/Proxy_O$ with the same values of Request-URI, From, To, and CallID, and any Dcs-State headers received for this call attempt.

MTA_O stores the Dcs-Media-Authorization, Dcs-State headers, Dcs-Remote-Party-ID header, Contact header and the SDP description for the duration of the call. The Dcs-State header values received in the 183-Session-Progress MUST be included in all requests and responses sent to the proxy for this call leg. The Dcs-Media-Authorization header value MUST be used for resource allocation. The Dcs-Remote-Party-ID header value is used for mid-call changes, such as transfer and three-way-calling.

MTA_O MUST send a PRACK to acknowledge receipt of the 183-Session-Progress. The PRACK message MUST be sent directly to the address specified in the Contact header of the received 183-Session-Progress.

An SDP MUST be included in the PRACK. The SDP in the PRACK MUST include a media (m=) line with a single CODEC to be used for this connection.

PRACK: (MTA_O -> MTA_T) **Header:**	***Requirement at MTA for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header received in the 183-Session-Progress*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
From:	*MUST be present.* *MUST be copies of same headers in the provisional response.*
To:	
Call-ID:	
Cseq: n_o+1 PRACK	*Sequence number MUST be one higher than previous sequence number, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.

v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after processing the SDP returned by the terminating MTA, and MUST contain a single CODEC choice.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK. The 200-OK response MUST be as follows.

200-OK: (MTA_T -> MTA_O) **Header:**	***Requirement on MTA for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.* *Method in Cseq MUST be PRACK.*
To:	
Call-ID:	
Cseq:	

Following receipt of the 183-Session-Progress response, MTA_O attempts to reserve access network resources based on the SDP parameters sent in the PRACK message. After successful completion of the resource reservation, MTA_O MUST send a PRECONDITION-MET message to MTA_T. This informs MTA_T that resources are available and that it may proceed and alert the end user. The PRECONDITION-MET message MUST be as follows.

PRECONDITION-MET: (MTA_O -> MTA_T) **Header:**	***Requirement at MTA for message generation***
PRECONDITION-MET SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. The value of the SIP-URL MUST be the Contact header received in the 183-Session-Progress*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE.*
To:	
Call-ID:	
Cseq: n_o+2 PRECONDITION-MET	*Sequence number MUST be one higher than the last sequence number sent by MTA_O, method MUST indicate PRECONDITION-MET*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after performing the QoS preconditions.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T- direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The originating MTA SHOULD be prepared to receive bearer channel packets once it has transmitted the PRECONDITION-MET.

The 200-OK response to the PRECONDITION-MET MUST be as follows.

200-OK: (MTA_T -> MTA_O) Header:	***Requirement on MTA for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRECONDITION-MET message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRECONDITION-MET message.*
To:	
Call-ID:	
Cseq:	

If the resource reservation fails, MTA_O SHOULD send a CANCEL to MTA_T via $CMS/Proxy_O$.

CANCEL: (MTA_O -> $CMS/Proxy_O$) Header:	***Requirement at MTA for message generation***
CANCEL DCS-URL SIP/2.0	*MUST be present. Method MUST be CANCEL. The value of the DCS-URL MUST be the same as was in the initial INVITE.*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
Dcs-State:	*MUST be present. MUST be copied from the 183-Session-Progress*
From:	*MUST be present.* *MUST be copies of same headers in INVITE.*
To:	
Call-ID:	
Cseq:	*Sequence number MUST be one higher than the last sequence number sent by MTA_O, method MUST indicate CANCEL*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The 200-OK response to the CANCEL MUST be as follows.

200-OK: ($CMS/Proxy_O$ -> MTA_O) Header:	***Requirement on MTA for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the CANCEL message*
From:	*From, To, CallID, and Cseq headers MUST match those of the CANCEL message.*
To:	
Call-ID:	
Cseq:	

6.4.4 MTA_T Receives Acknowledgement of 183-Session-Progress

After sending the 183-Session-Progress response to the INVITE, MTA_T MUST wait for the PRACK message acknowledging the Session-Progress. The PRACK message headers MUST be checked as follows.

PRACK: (MTA_O -> MTA_T) Header:	***Requirement at MTA for message checking***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header sent in the 183-Session-Progress*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE request.*
To:	
Call-ID:	
Cseq: n_o+1 PRACK	*Sequence number MUST be one higher than sequence number in INVITE, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after processing the SDP returned by the terminating MTA, and MUST contain a single CODEC choice.*

On receipt of this PRACK, MTA_T MUST respond with a 200-OK. The 200-OK response MUST be as follows.

200-OK: (MTA_T -> MTA_O) Header:	***Requirement on MTA for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	
Cseq:	

Following receipt of the PRACK message, MTA_T attempts to reserve access network resources based on the SDP parameters received in the PRACK message, or based on the SDP parameters received in the INVITE if no SDP was present in the PRACK..

After MTA_O successfully completes the resource reservation, it sends a PRECONDITION-MET message to MTA_T. This informs MTA_T that resources are available and that it may proceed and alert the end user. MTA_T MUST check and verify the PRECONDITION-MET message as follows.

PRECONDITION-MET: (MTA_O -> MTA_T) Header:	***Requirement at MTA for message checking***
PRECONDITION-MET SIP-URL SIP/2.0	*MUST be present. Method MUST be PRECONDITION-MET. The value of the SIP-URL MUST be the Contact header sent in the 183-Session-Progress*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE request.*
To:	
Call-ID:	
Cseq: n_o+1 PRECONDITION-MET	*Sequence number 'n_o+1' MUST be one higher than the last sequence number sent by MTA_O, method MUST indicate PRECONDITION-MET*

Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after performing the preconditions.*

MTA_T MUST respond to the PRECONDITION-MET request with a 200-OK. The 200-OK response to the PRECONDITION-MET MUST be as follows.

200-OK: (MTA_T -> MTA_O) **Header:**	***Requirement on MTA for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRECONDITION-MET message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRECONDITION-MET message.*
To:	
Call-ID:	
Cseq:	

On receipt of the PRECONDITION-MET message, and successfully reserving the network resources needed for its media flows, MTA_T MUST cancel timer T3, and continue with the alerting procedures of section 6.4.5.

If the resource reservation fails, MTA_T MUST send a 580-Precondition-Failure response to $CMS/Proxy_T$.

580-Precondition-Failure: **(MTA_T -> $CMS/Proxy_T$)** **Header:**	***Requirement on MTA for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*MUST be present. MUST be copied from the INVITE message.*
From:	*From, To, CallID, and Cseq headers MUST match those of the INVITE message.*
To:	
Call-ID:	
Cseq:	

6.4.5 MTA_T sends 180-Ringing

Once MTA_T receives the PRECONDITION-MET message, and resource reservation is successful, the MTA continues with normal call processing. If the destination is immediately ready to accept the connection, without any alerting needed, MTA_T continues with the procedures given in section 6.4.7.

If the destination endpoint is not immediately ready to accept the connection, MTA_T MUST send a 180-RINGING message to $CMS/Proxy_T$. This response is a second provisional response to the initial INVITE, and is sent through the proxies.

180 Ringing: (MTA_T -> $CMS/Proxy_T$) Header:	***Requirement***
SIP/2.0 180 Ringing	*Status line with status code 180 MUST be present.*
Via:	*MUST be present and copied from INVITE message.*
Dcs-State:	*MUST be present and copied from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the received INVITE. This triple identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present. MUST be same as in 183-Session-Progress*
Cseq:	*MUST be present. MUST be the same as that in the received INVITE. Method MUST be INVITE. Identifies the message which caused this response.*
Rseq: x_t+1	*MUST be present. MUST be value one greater than most recent provisional response Rseq value*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of PRACK.

On sending the status 180-RINGING message, the terminating MTA MUST start the transaction timer (T3) with value T-ringing. The default value of (T-ringing) is given in Appendix A. Timer T3 is canceled by the user indicating call acceptance, or receipt of a BYE or CANCEL request. On expiration of timer T3, MTA_T MUST either perform call-forwarding-no-answer, or sends a 480-Temporarily-Unavailable response to MTA_O.

After sending the 180-Ringing response to the INVITE, MTA_T MUST wait for the PRACK message acknowledging the response. The PRACK message headers MUST be checked as follows.

PRACK: (MTA_O -> MTA_T) Header:	***Requirement at MTA for message checking***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header sent in the 183-Session-Progress*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE request.*
To:	
Call-ID:	
Cseq: n_o+1 PRACK	*Sequence number 'n_o+1' MUST be one higher than sequence number in previous request, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 180-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*

On receipt of this PRACK, MTA_T MUST respond with a 200-OK. The 200-OK response MUST be as follows.

200-OK: (MTA_T -> CMS/Proxy) Header:	***Requirement on MTA for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	
Cseq:	

6.4.6 MTA_O receives 180-Ringing/183-Media

After completing the resource reservation, and sending the PRECONDITION-MET message to MTA_T, MTA_O will receive either (1) a provisional response of 180-Ringing or 183-Session-Progress(Media), (2) a final response of 200-OK or (3) a client error. This section covers the procedures for the provisional responses, 180 and 183, and section 6.4.8 covers the procedures for the final responses.

MTA_O MUST verify the headers of the provisional response according to the following table.

18x Provisional Response: ($CMS/Proxy_O$ -> MTA_O) Header:	***Requirement at CMS/Proxy for message generation Requirement for checking at MTA***
SIP/2.0 180 Ringing Or SIP/2.0 183 Session Progress	*Status line with status code 180 or 183 MUST be present.*
Via:	*MUST be present and match that in INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST match those in the initial INVITE. This triple identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present. MUST be same as in 183-Session-Progress*
Cseq: n_o INVITE	*MUST be present. MUST be the same as that in the initial INVITE. Method MUST be INVITE. Identifies the message which caused this response.*
Session: Media	*MUST be present for 183-Session-Progress, MUST contain 'Media' MUST NOT be present for 180-Ringing*
Rseq: x_t+1	*MUST be present. MUST be value one greater than most recent provisional response Rseq value*

Upon receipt of the 18x message, MTA_O MUST stop the T3 session timer, and restart the session timer T3 with the value T-ringback. The default value of (T-ringback) is given in Appendix A. The T3 timer MUST be cancelled on receipt of 200-OK or other final response.

The 180-Ringing response indicates to MTA_O that it SHOULD supply a local ringback. The 183-Session-Progress response indicates that the ringback is supplied via audio packets from the data network, and that MTA_O SHOULD enable the media receive path.

MTA_O MUST acknowledge the 18x provisional response with a PRACK message, as described in the following table.

PRACK: (MTA_O -> MTA_T) Header:	***Requirement at MTA for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header received in the 183-Session-Progress*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in Request from $CMS/Proxy_O$.*
Call-ID:	
Cseq: n_o+3 ACK	*Sequence number MUST be one higher than sequence number in the latest request, method MUST indicate PRACK*
Rack: x_t+1 n_o INVITE	*Value 'x_t' MUST be a copy of the value in the Rseq header of the 18x-Session-Progress. Value 'n_o' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T- direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK. The 200-OK response MUST be as follows.

200-OK: (MTA_T -> MTA_O) Header:	***Requirement on MTA for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	
Cseq:	

6.4.7 MTA_T Sending final Response

After MTA_T has successfully reserved resources, and received the PRECONDITION-MET message from MTA_O indicating it had also successfully reserved resources, it performs whatever alerting procedures are required and signals when the MTA is ready to begin media transfers. For a typical telephony service, this is indicated by the user 'going offhook' and 'answering the phone.' The case of a successful completion of a call is covered in section 6.4.7.1, and the various error cases are covered in section 6.4.7.2 and 6.4.7.3.

6.4.7.1 MTA_T Sending 200-OK

Once MTA_T determines that the user is willing to accept the incoming call (e.g. off-hook or hook-flash), the session establishment is considered complete. If there are any Dcs-Also headers or Dcs-Replaces headers in the INVITE request, they MUST be checked prior to sending the 200-OK, as specified in Section 6.8.6. MTA_T MUST send a 200-OK status message to MTA_O, via $CMS/Proxy_T$.

200-OK: (MTA_T -> $CMS/Proxy_T$) Header:	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
Dcs-State:	*MUST be present, copy of value from INVITE message*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the received INVITE. Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE.*

On sending the 200-OK, MTA_T MUST stop timer T3. If necessary, MTA_T MUST also commit to resources that have been reserved for this call and MAY begin sending bearer channel packets.

MTA_T SHOULD be prepared to receive bearer channel packets once it has sent the final response.

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

The ACK message header fields MUST be verified as follows:

ACK: (MTA_O -> MTA_T) Header:	***Requirement at MTA for checking message***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value sent in the Contact header in the 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present. MUST be copies of same headers in initial INVITE request.*

To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

6.4.7.2 MTA$_T$ sending 3xx-Redirect

If MTA$_T$ wishes to forward the call (e.g. if call-forwarding-no-answer is enabled at MTA$_T$), a 302-Redirect status response MUST be sent to CMS/Proxy$_T$ with the forwarded-to destination URI in the contact header.

302-Redirect: (MTA$_T$ -> CMS/Proxy$_T$) Header:	***Requirement on MTA for message generation Requirement on CMS/Proxy for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST be inserted by MTA$_T$. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*MUST be copied from the INVITE message*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be inserted by MTA$_T$ and carries the new destination information. It MUST be a valid URI.* *If the new destination is a telephone number, then the format of the URI MUST be a tel: URI where the URI contains the sequence of dialed digits, including any prefixes.*
Expires:	*MAY be present*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

If MTA$_T$ is not a subscriber to the Call Forwarding service, or if any of the header verification checks fail, CMS/Proxy$_T$ MUST send a 480-Temporarily-Unavailable error response to CMS$_O$, and MUST send a CANCEL to MTA$_T$. Otherwise, CMS/Proxy$_T$ MUST send an ACK message to MTA$_T$. The required fields of the message are as shown below. The transaction between MTA$_T$ and CMS/Proxy$_T$ is now complete.

ACK: (CMS/Proxy$_T$ -> MTA$_T$) Header:	***Requirement at CMS/Proxy***
ACK SIP-URL SIP/2.0	*The Response line MUST be present.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS/Proxy$_T$.*
From:	*MUST be present.* *MUST be copies of same headers in Request from CMS/Proxy$_O$.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in response from MTA$_T$, method MUST indicate ACK*

6.4.7.3 MTA$_T$ Sending Other Status Response to INVITE request

A final error response, 4xx, 5xx, or 6xx response, MUST be sent as per [11]. This includes, but is not limited to, 480-Temporarily-Unavailable. The error response MUST be generated as follows.

Error: (MTA_T -> $CMS/Proxy_T$) Header:	***Requirement on MTA for message generation Requirement on CMS/Proxy for message checking***
SIP/2.0 xxx	*Status line header MUST be inserted by MTA_T. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*MUST be copied from the INVITE message*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

$CMS/Proxy_T$ MUST send an ACK message to acknowledge the error response. The transaction between MTA_T and $CMS/Proxy_T$ is now complete.

ACK: ($CMS/Proxy_T$ -> MTA_T) Header:	***Requirement at CMS/Proxy for message generation Requirement at MTA for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ACK. Request-URI MUST be copy of initial INVITE message*
Via:	*MUST be present. MUST be the IP address or FQDN of $CMS/Proxy_T$.*
From:	*MUST be present. MUST be copies of same headers in initial INVITE message.*
To:	
Call-ID:	
Cseq: n_0 ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE message. Method MUST indicate ACK*

6.4.8 MTA_O Receives final response from MTA_T

6.4.8.1 MTA_O Receiving 200-OK

Once the terminating endpoint determines that the user is willing to accept the incoming call (e.g. off-hook or hook-flash), it sends a 200-OK status message to MTA_O, via the CMS/Proxies. The message sent by $CMS/Proxy_O$ to MTA_O MUST be as follows.

200-OK: ($CMS/Proxy_O$ -> MTA_O) Header:	***Requirement for $CMS/Proxy_O$ for message generation Requirement for MTA_O for message checking***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be identical to the initial INVITE message. Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE.*

On receiving the final response, MTA_O MUST stop timer T3. MTA_O MUST also commit to resources that have been reserved for this call and SHOULD begin sending bearer channel packets.

MTA_O MUST acknowledge the 200-OK response with an ACK message. The header fields MUST be generated as follows:

ACK: (MTA_O -> MTA_T) **Header:**	***Requirement at MTA_O for message generation***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the Contact header in the 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in initial INVITE request.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

6.4.8.2 MTA_O receiving 3xx-Redirect

If the terminating endpoint wished to forward the call (e.g. if call-forwarding-no-answer was enabled at the destination), a 302-Redirect status response is sent back through the CMS/Proxies with the forwarded-to destination URI in the contact header. The message sent by $CMS/Proxy_O$ to MTA_O MUST be as follows.

302-Redirect: ($CMS/Proxy_O$ -> MTA_O) **Header:**	***Requirement on CMS/Proxy for message generation*** ***Requirement on MTA for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be identical to the INVITE message*
From:	*From, To, CallID, and Cseq headers MUST be identical to the INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be present. Carries the new destination information. MUST be a valid URI.* *MUST have a private-param url-parameter, as described in 7.6.8.2.*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

MTA_O MUST send an ACK message to $CMS/Proxy_O$. The required fields of the message are as shown below.

ACK : (MTA_O -> $CMS/Proxy_O$) **Header:**	***Requirement at CMS/Proxy***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the Contact header in the 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in initial INVITE request.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

In response to a 302-Redirect final response, MTA_O SHOULD initiate a new INVITE, as described in section 6.4.1, with the Request-URI being the value received in the Contact header of the 302-Redirect response. Typically this URI will include a url-parameter indicating 'private', and will be honored by $CMS/Proxy_O$ for only a short period of time to initiate a call.

6.4.8.3 MTA_O receiving other error response

A final error response, 4xx, 5xx, or 6xx response, MAY be sent as per [11]. This includes, but is not limited to, 480-Temporarily-Unavailable. The error response MUST be generated and verified as follows.

Error: ($CMS/Proxy_O$ -> MTA_O) Header:	***Requirement on CMS/Proxy for message generation Requirement on MTA for message checking***
SIP/2.0 xxx	*Status line header MUST be inserted by $CMS/Proxy_O$. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

MTA_O MUST send an ACK message to acknowledge the error response.

ACK: (MTA_O -> $CMS/Proxy_O$) Header:	***Requirement at MTA for message generation Requirement at CMS/Proxy for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ACK. Request-URI MUST be copy of initial INVITE message.*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
Dcs-State:	*MUST be present. MUST be the value(s) received for this call leg*
From:	*MUST be present.* *MUST be copies of same headers in initial INVITE message.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE message. Method MUST indicate ACK*

6.4.9 Session Timer expiration at MTA_O

On expiration of timer T3, MTA_O SHOULD send a CANCEL request to MTA_T through the CMS/Proxies, and MUST release all resources reserved for this connection. The CANCEL request MUST be as described below.

CANCEL: (MTA_O -> $CMS/Proxy_O$) Header:	***Requirement at MTA for message generation***
CANCEL DCS-URL SIP/2.0	*MUST be present. Method MUST be CANCEL. The value of the DCS-URL MUST be the same as the initial INVITE message.*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
Dcs-State:	*MUST be present. MUST be the value(s) received for this call leg.*
From:	*MUST be present.* *MUST be copies of same headers in Request from $CMS/Proxy_O$.*
To:	
Call-ID:	
Cseq: n_o+1 CANCEL	*Sequence number 'n_o+1' MUST be one higher than the last sequence number sent by MTA_O, method MUST indicate CANCEL*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The 200-OK response to the CANCEL MUST be as follows.

200-OK: (CMS/Proxy$_O$ -> MTA$_O$) Header:	***Requirement on MTA for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the CANCEL message*
From:	*From, To, CallID, and Cseq headers MUST match those of the CANCEL message.*
To:	
Call-ID:	
Cseq:	

6.5 Initiating a Call Return

INVITE(return-call) is used to initiate a callback, with the MTA providing the Remote-Party-ID URL from the call that the customer wishes to return. This is done in the current PSTN network by the customer dialing *69 to return the most recent call. To emulate the current PSTN *69 behavior, the MTA saves and returns the Remote-Party-ID URL from the most recently received INVITE with a new Call-ID. However, the DCS Return-Call function can be used to re-place any call, whether originated or received or transferred.

Upon receiving a user request for call-return (e.g. *69 in the PSTN), the originating MTA (which was the terminating MTA of the previous call) MUST initiate a call by sending an INVITE(return-call) message to its CMS/Proxy. INVITE(return-call) messages are like INVITE messages. They differ from the INVITE in the Request-Line.

The To: header, without any requirement of the originating user for privacy, would contain the dialed digits, e.g., tel:*69. If the originating user requested privacy, the To: header would be a random string. The Proxy uses the Request-URI to determine the proper handling of the call.

INVITE (return-call): (MTA$_O$ -> CMS/Proxy): Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*The Request URI MUST be a DCS-URL.* *The URL MAY include a private-param which MUST be the encrypted Remote-Endpoint-ID from a previous call.* *The URL MAY be a tel: URL (which typically occurs when the subscriber has Caller-ID service and the previous call originator did not request privacy).*
--all other headers--	*All other headers are unchanged from INVITE. See 6.4.1.*

MTA$_O$ MUST follow the message generation and retransmission rules as given for INVITE described in Section 6.4.

CMS/Proxy$_O$ converts this INVITE into an INVITE identical to that described in Section 6.4. Processing of the call is otherwise identical.

6.6 Initiating a Call Trace

Invite(call-trace) is used to request a trace of a previously received call, to report the call to law enforcement as an obscene or harassing call. This is done in the current PSTN network by the customer dialing *57 to report the most recent call. To emulate the current PSTN *57 behavior, the MTA saves and returns the Remote-Party-ID URL string from the most recently received INVITE with a new Call-ID. However, the DCS Call-Trace function can be used to report any call, whether originated or received or transferred.

Upon receiving a user request for call-trace (e.g. *57 in the PSTN), the originating MTA (which was the terminating MTA of the previous call) MUST initiate a call by sending an INVITE(call-trace) message to its CMS/Proxy. INVITE(call-trace) messages are like INVITE messages. They differ from the INVITE in the Request-Line and by the addition of a Dcs-Trace-Party-ID header.

INVITE (call-trace): (MTA_O-> CMS/Proxy): Header:	***Requirement***
INVITE sip: call-trace@Host(dp-o) SIP/2.0	*The Request URI MUST have username of "call-trace" and hostname identifying CMS/Proxy$_O$.*
Dcs-Trace-Party-ID: URL	*MUST be present. MUST contain the URL from the previous Dcs-Remote-Party-ID header*
--all other headers--	*All other headers are unchanged from INVITE*

MTA_O MUST follow the retransmission rules as given for INVITE described in Section 6.4.

$CMS/Proxy_O$ records the complaint, then converts this INVITE into an INVITE identical to that described in Section 6.4. The call completes either to the Service Provider's office (to obtain further information about the complaint) or to an announcement server telling the customer to call the Service Provider during normal business hours to give the further information.

6.7 Initiating a 9-1-1 call

A call for emergency services, e.g. 9-1-1, MUST follow the procedures given for a basic call, as given in section 6.4, with the following exceptions[4].

MTA_O SHOULD disable the call waiting feature, so that any incoming call to MTA_O is given a BUSY error instead of call-waiting treatment.

If MTA_O detects a desire to terminate the conversation, MTA_O SHOULD NOT sent a BYE request to the Emergency Services Center; rather MTA_O SHOULD wait for a BYE request to initiate at the Emergency Services Center.

6.8 SIP Messages during an active call

The messages in this section change the characteristics of an active call. Examples include call transfer, call-transfer to bridge for three-way-calling, or change in a call's SDP description when a call is put on hold.

[4] Due to the fact that the MTA is considered untrusted in the DCS architecture, and therefore is under total control of the subscriber, the requirements in this section are given as SHOULDs, rather than MUSTs.

Dcs-Also and Dcs-Replaces provide tools by which many call control services are built. For purposes of this specification, only three are completely specified at the initiator: blind transfer, consultative transfer, and ad-hoc conferencing. The procedures necessary to support these are completely specified at the recipient. Based on knowledge of the recipient behavior, the originator MAY perform many other complex call control operations, beyond those specified here.

6.8.1 Initiating Call Hold: INVITE(hold)

To place a call on hold, an INVITE(hold) message is sent directly to the MTA that is to be put on hold. It is a standard SIP INVITE message, with the IP address in the connection field in SDP ("c=") set to 0.0.0.0. The format of the INVITE message sent by the initiating MTA (MTA_I) and the requirements on the header fields checked at the receiving MTA (MTA_R) are as follows.

INVITE(Hold): (MTA_I -> MTA_R) **Header:**	***Requirements on MTA_I for message generation*** ***Requirements on MTA_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the Contact header from the INVITE message or 183-Session-Progress response for this call.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present.* *MUST contain the IP address or FQDN of the initiating MTA.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the hold is initiated by the called party.*
To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 6.1.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST be set to 0.0.0.0*

On receiving an INVITE(hold), MTA_R MUST send the 200-OK with the updated SDP description to MTA_I, and stop sending bearer channel packets to that same party.

200-OK: (MTA_R -> MTA_I) **Header:**	***Requirement for MTA_R for message generation*** ***Requirement for MTA_I for message checking***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(Hold) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(Hold).*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.

v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST be set to 0.0.0.0*

MTA_I sends an ACK to MTA_R. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

ACK: (MTA_I -> MTA_R) **Header:**	***Requirement at MTA_I for message generation*** ***Requirement at MTA_R for message checking***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the Contact header in the Initial INVITE or initial 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(Hold) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(Hold) request, method MUST indicate ACK*

6.8.2 Resuming a held call: INVITE(resume)

The MTA that placed the call on hold MUST be the one to take it off hold. To take a call off hold, an INVITE(resume) is sent. An INVITE(resume) is an INVITE(hold) message with the SDP description of the call being reinstated. The format of the INVITE message sent by the initiating MTA (MTA_I) and the requirements on the header fields checked at the receiving MTA (MTA_R) are as follows.

INVITE(Resume): (MTA_I -> MTA_R) **Header:**	***Requirements on MTA_I for message generation*** ***Requirements on MTA_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the Contact header from the INVITE message or 183-Session-Progress response for this call.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present.* *MUST contain the IP address or FQDN of the initiating MTA.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the hold is initiated by the called party.*
To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 6.1.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST NOT be set to 0.0.0.0*

If the MTA that receives the resume is willing to reinstate the bearer channel, it MUST update the SDP description for the call and send a 200-OK with the saved SDP description for the active call. If not, it MUST send a 4xx (client error) response. The 200-OK response MUST be as follows:

200-OK: (MTA_R -> MTA_I) **Header:**	***Requirement for MTA_R for message generation*** ***Requirement for MTA_I for message checking***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(Resume) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(Resume).*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST NOT be set to 0.0.0.0*

MTA_I sends an ACK to MTA_R. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

ACK: (MTA_I -> MTA_R) **Header:**	***Requirement at MTA_I for message generation*** ***Requirement at MTA_R for message checking***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the Contact header in the Initial INVITE or initial 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(Resume) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(Resume) request, method MUST indicate ACK*

6.8.3 Initiating Blind Call Transfer

Two types of call transfer are described in this specification. Blind transfer is when the party initiating the transfer sequence does not have an active connection to the desired new destination. Therefore, the transferring party has no assurance that the call transfer will be successful. Consultative transfer is when the party initiating the transfer sequence has an active connection to the desired new destination. Both are realized with a combination of Dcs-Also and Dcs-Replaces headers in an INVITE request. This section describes only blind transfer; the next section describes consultative transfer.

By initiating a blind call transfer, the initiator is agreeing to be billed for a logical call-leg from himself to the new destination for the duration of the transferred call.

An INVITE request containing both a Dcs-Also header and a Dcs-Replaces header is used to transfer a call in progress. The "Dcs-Also:" header indicates the new party to be added and the "Dcs-Replaces:" header indicates the party to be removed. This INVITE is referred to as INVITE(also,replace).

The INVITE request for a blind transfer MUST be sent by an MTA to its CMS/Proxy. The REQUEST URI MUST be the value of the Contact header from either the INVITE message (if MTA is the destination of the call) or from the 183-Session-Progress (if MTA is the originator of the call).

The call-leg identification (From, To, and Call-ID) MUST match an active call. If the call originator is initiating the blind transfer, From and To will match those in the initial INVITE. If the call destination is initiating the blind transfer, the values of From and To will be reversed.

The Dcs-Also header MUST contain the Dcs-URL of the desired destination. This Dcs-URL MAY contain a private-param. Additional header parameters (e.g. Dcs-Also: URL ? header=value & header=value) SHOULD NOT be attached to the Dcs-Also header, which distinguishes this from the Consultative transfer described later.

The Dcs-Replaces header MUST contain the same value as the From header, which refers to the initiating MTA.

The INVITE request MUST include all the Dcs-State headers given to the MTA by its proxy with matching call-leg identification (From, To, Call-ID).

The INVITE request MUST NOT contain an SDP description.

The requirements on the headers which the MTA MUST include in the message are shown below:

INVITE(also,replace): (MTA_I->$CMS/Proxy_I$) Header:	***Requirement on MTA for message generation Requirement on CMS/Proxy for message checking***
INVITE SIP-URL SIP/2.0	*MUST be present. The Request-URI MUST be a SIP-URL as defined in Section 4.2, and MUST be a copy of the Contact header from the initial Initial INVITE or initial 183-Session-Progress message.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present and MUST be the address of the originator of this message. Typically, the terminating MTA of the active call originates the INVITE(also,replace).*
Dcs-Remote-Party-ID:	*SHOULD be present.*
Dcs-Also: DCS-URL	*MUST be present and identifies the new address of the destination to which the recipient of this INVITE(also,replace) is to issue an INVITE. Identifies new call leg to be created. It MAY be any valid DCS-URL. It MAY include a private-param in a DCS-URL.*
Dcs-State:	*MUST be present, and contain the Dcs-State header given by the proxy with matching call-leg identification (From, To, Call-ID)*
From:	*MUST be either the From or To header of the current call being transferred, whichever is the address of the originator of this message. Typically, INVITE(also,replace) is sent by the call-ed party, and therefore this is the To header.*
To:	*MUST be either the From or To header of the current call being transferred, whichever is the address of the destination of this message. Typically, this is the call-ing party, and therefore this is the From header*
Call-ID: ID	*The Call-ID MUST be the same as the Call-ID of the active call*
CSeq: n_i INVITE	*MUST be as defined in 6.1.*
Dcs-Replaces:	*This identifies the call-leg to be torn down at the endpoint receiving the INVITE(also,replace). It MUST be identical to the value of the From header in this message.*

A call flow illustrating the use of INVITE(also,replace) in blind call-transfer is shown in Appendix N.

The INVITE(also,replace) traverses through CMS/Proxies to the destination. The INVITE(also,replace) received at the MTA from its CMS/Proxy and requirements on its headers is shown in Section 6.8.6.

6.8.4 Initiating Consultative Call Transfer

Both consultative and blind transfer are realized with a combination of Dcs-Also and Dcs-Replaces headers in an INVITE request. This section describes only consultative transfer; the previous section described blind transfer.

Consultative transfer is when the party initiating the transfer sequence (the initiator) has an active connection to the client (the client), and also has an active connection to the desired new destination (the consultant). Typically the client had previously called the initiator; then the initiator called the consultant and decided to transfer the client to the consultant.

By initiating a consultative call transfer, the initiator is agreeing to be billed for a logical call-leg from himself to the consultant for the duration of the transferred call. If the client had initially called the initiator, then the billing of the resulting transferred call will be split between the client and the initiator. If the initiator had initially called the client, then the billing of the resulting transferred call will be entirely to the initiator.

An INVITE request containing both a Dcs-Also header and a Dcs-Replaces header is used to transfer a call in progress. The “Dcs-Also:” header indicates the new party to be added and the “Dcs-Replaces:” header indicates the party to be removed. This INVITE is referred to as INVITE(also,replace).

The INVITE request for a consultative transfer MUST be sent by an MTA to its CMS/Proxy, to be forwarded to the consultant. The REQUEST URI MUST be the value of the Contact header from either the INVITE message (if MTA is the destination of the call from the consultant) or from the 183-Session-Progress (if MTA is the originator of the call to the consultant).

The call-leg identification (From, To, and Call-ID) MUST match the active call with the consultant. If the call originator is initiating the consultative transfer, From and To will match those in the initial INVITE. If the call destination is initiating the consultative transfer, the values of From and To will be reversed.

The Dcs-Also header MUST contain the Dcs-URL of the client, as received by the initiator in the Dcs-Remote-Party-ID header. This Dcs-URL MAY contain a private-param. An additional header parameter (e.g. Dcs-Also: URL ? header=value & header=value & header=value) MUST be attached with “Call-ID=” and the value of the Call-ID for the existing call between the initiator and the client. An additional header parameter MUST be attached with “Dcs-Replaces=” and the value of either From or To of the call-leg identification (whichever refers to the initiator) of the call between the initiator and the client. An additional header parameter MUST be attached with “Dcs-State=” and the value(s) of the state header for the call between the initiator and the client. Other additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The Dcs-Replaces header MUST contain the same value as the From header, which refers to the initiating MTA.

The INVITE request MUST include the Dcs-State header given to the MTA by its proxy with matching call-leg identification (From, To, Call-ID).

The INVITE request MUST NOT contain an SDP description.

The requirements on the headers which the MTA MUST include in the message are shown below:

INVITE(also,replace): (MTA_I->$CMS/Proxy_I$) Header:	***Requirement on MTA for message generation Requirement on CMS/Proxy for message checking***
INVITE SIP-URL SIP/2.0	*MUST be present.* *The Request-URI MUST be a SIP-URL as defined in Section 4.2.* *Request-URI MUST be from a Contact header from Initial INVITE or initial 183 of the active call with the consultant.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present and MUST be the address of the originator of this message. Typically, the terminating MTA of the active call originates the INVITE(also,replace).*
Dcs-Also: DCS-URL ? Call-ID=ID & Dcs-Replaces=URL & Dcs-State=DS	*MUST be present and identifies the client.* *MAY be any valid DCS-URL.* *MUST be copied from the Dcs-Remote-Party-ID of the call with client.* *MAY include a private-param in a DCS-URL.* *Call-ID and Dcs-Replaces and Dcs-State attached headers MUST be present, and be as described above.*
Dcs-State:	*MUST be present, and contain the Dcs-State header given by the proxy with matching call-leg identification (From, To, Call-ID)*
From:	*MUST be either the From or To header of the call with consultant, whichever is the address of the originator of this message. Typically, consultative transfer is sent by the calling party, and therefore this is the From header.*
To:	*MUST be either the From or To header of the call with consultant, whichever is the address of the destination of this message. Typically, this is the call-ed party, and therefore this is the To header*
Call-ID: ID	*The Call-ID MUST be the same as the Call-ID of the active call with consulotant*
CSeq: n_i INVITE	*MUST be as defined in 6.1.*
Dcs-Replaces:	*MUST be identical to the value of the From header in this message.*

A call flows illustrating the use of INVITE(also,replace) in consultative call-transfer is shown in Appendix O.

The INVITE(also,replace) traverses through CMS/Proxies to the destination. The INVITE(also,replace) received at MTA_R from $CMS/Proxy_R$ and requirements on its headers is shown in Section 6.8.6.

6.8.5 Initiating an Ad-hoc Conference

An ad-hoc conference is formed when an initiator has two simultaneous active calls, one to party A and one to party B, and desires to connect them together. While it is possible to do this locally, within the MTA and without the knowledge of the proxies, it consumes double the access network resources of a conference bridge and is therefore discouraged. The MTA SHOULD NOT implement local bridging of multiple calls.

When creating a call with multiple parties connected to a single destination, e.g. to a bridge for an ad-hoc conference, it is often more convenient and efficient to request the destination to initiate the additional calls, rather than initiate a call transfer to direct each party to the desired new destination.

An INVITE request containing a Dcs-Also header is used to initiate an ad-hoc conference. The "Dcs-Also" header indicates the party to be added to the conference. This INVITE is referred to as INVITE(also).

Ad-hoc conference initiation involves establishing a new connection from the conference initiator to the bridge service. This new connection has all the properties of a normal call, and the INVITE message MUST contain all the header fields as described in 6.4.1. The INVITE request for an ad-hoc conference MUST be sent by an MTA to its CMS/Proxy, to be forwarded to the bridge service.

An MTA that sends INVITE(Also) is indicating a willingness to pay for the additional call segments between itself and the bridge for all of the parties in the conference. Proper billing arrangements are established by the CMS/Proxy.

When an ad-hoc conference is initiated, and a bridge server URI is provisioned in the MTA, then Request-URI MUST be the provisioned URI. If no bridge server is provisioned, then the Request-URI MUST be sip:bridge@Host(dp-o). The username "bridge" is therefore reserved for this purpose.

The INVITE(Also) message sent by the MTA to initiate an ad-hoc conference MUST be as follows:

INVITE (also): (MTA_O -> $CMS/Proxy_O$): Header:	***Requirement***
INVITE sip:bridge@Host(dp-o) SIP/2.0	*The Request URI MUST be a DCS-URL, as specified above*
--all other headers--	*All other headers are unchanged from INVITE*
Dcs-Also: DCS-URL ? Call-ID=ID-A & Dcs-Replaces=URL-A	*See requirements below.*
Dcs-Also: DCS-URL ? Call-ID=ID-B & Dcs-Replaces=URL-B	*See requirements below.*
	An empty line MUST be present between the headers and the message body
--SDP description--	*MUST be an SDP description, as specified in 6.4.1*

The INVITE(also) request initiating an ad-hoc conference MUST contain two or more Dcs-Also headers, one header for each participant in the conference.

For each participant in the ad-hoc conference (referred to as party-X in this paragraph), the Dcs-Also: header MUST contain the DCS-URL obtained from the Dcs-Remote-Party-ID header of the INVITE message or 183-Session-Progress for the active call with party-X. This MAY be a DCS-URL containing a private-param. An additional header parameter (e.g. Dcs-Also: URL ? header=value & header=value) MUST be attached with "Call-ID=" and the value of the Call-ID for the existing call between the initiator and party-X. An additional header parameter MUST be attached with "Dcs-Replaces=" and the value of either From or To of the call-leg identification (whichever refers to the initiator) of the call between the initiator and party-X. An additional header parameter MUST be attached with "Dcs-State=" and the value(s) of the state headers for the call between the initiator and party-X. Other additional header parameters SHOULD NOT be attached to the Dcs-Also header.

A call flow illustrating the use of INVITE(also) in establishing an ad-hoc conference is shown in Appendix P.

The INVITE(Also) sent by MTA_O is sent to $CMS/Proxy_O$ which then identifies the CMS associated with the destination. The CMS associated with the destination receives the Dcs-Billing-Info and Dcs-Billing-ID for each of the call legs to be transferred.

6.8.6 Call Control: Receipt of INVITE(also/replace)

This section describes the handling of INVITE(Also), INVITE(Replace), and INVITE(Also,replace), collectively referred to here as INVITE(also/replace). When both Dcs-Also headers and Dcs-Replaces headers are present in the same INVITE, the Dcs-Also headers are processed first, followed by the Dcs-Replaces headers.

Typical use of INVITE(also/replace) is for call features such as call-transfer and three-way-calling, where the other party in the call initiated the special feature and is requesting the receiving MTA to alter its current connections in support of that feature.

An MTA MUST be capable of receiving an INVITE(also/replace) message from its CMS/Proxy at any time during an active call. The INVITE(also/replace) message received at the MTA is shown in the table below. The Requirements shown for this message specify (1) the actions expected of the CMS/Proxies in processing the message from the originating MTA, and (2) the action required of the MTA if the field is not present or is not in the correct format.

INVITE(also/replace): (CMS/Proxy$_R$-> MTA$_R$): Header:	***Requirement***
INVITE sip: Host(mta-r) SIP/2.0	*MUST be present. MUST be sufficient for the MTA to determine the proper line being addressed.*
Via: SIP/2.0/UDP Host(dp-r), a	*MUST be present.*
Dcs-Also: DCS-URL	*If present, this header contains a DCS-URL, and MAY contain attached headers* *The URL MUST be encrypted by the CMS/Proxy's private key, with the url-parameter "private" included.* *The URL in the Also header is passed back to the CMS/Proxy in the Request-URI of the INVITE initiated by the recipient of this message to establish the new call-leg.*
From:	*Call identification {From, To, CallID}, and CSeq MUST be unchanged from that sent by MTA$_I$.*
To:	
Call-ID:	
CSeq: ni INVITE	
Dcs-Replaces:	*If present, this header contains a SIP-URL, and MAY contain attached headers.* *MUST matche either the From: header or To: header of a current call.* *Call-ID (or a Call-ID attached header) MUST match same current call.*

The INVITE(also/replace) MAY contain an SDP description, and, if so, indicates a new media session MUST be established before processing the Dcs-Also or Dcs-Replaces headers. This is used in this specification in the ad-hoc conference; the other services do not include a SDP. Procedures for establishing the media session are identical to Section 6.4.

MTA$_R$ MUST check the validity of the Dcs-Also and Dcs-Replaces headers. An MTA that is not capable of performing local bridging of media streams SHOULD reject an INVITE(also/replace) that would result in two or more call legs with the same Call-ID once all the Dcs-Also and Dcs-Replaces headers have been processed. For each Dcs-Replaces header, MTA$_R$ MUST verify it has a call active with the matching Call-ID and that the Dcs-Replaces value matches either the From or To call-leg identification

Once any media session is established, and MTA$_R$ has checked the validity of the Dcs-Also and Dcs-Replaces headers, MTA$_R$ MUST send the final response to the INVITE(also/replace). The final response to an INVITE(also/replace) is a 200-OK as in an INVITE.

On receiving an INVITE(also/replace) that includes the Dcs-Also: header, MTA$_R$ MUST send an INVITE message to CMS/Proxy$_R$ for each Dcs-Also header present. The Request-URI MUST be the DCS-URL from the Dcs-Also header of the received INVITE(also/replace), and the To: header MUST be generated by MTA$_R$ as a locally unique string. MTA$_R$ does not know where the call is being forwarded (although CMS/Proxy$_R$ knows this information when it decrypts the Request-URI). The To: header is therefore an identifier without any significance to the caller or the final destination of the INVITE. Any additional headers given with the Dcs-URL in the Dcs-Also: header MUST be copied into the INVITE.

INVITE: (MTA$_R$ ->CMS/Proxy$_R$) Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*The Request-URI is the DCS-URL from the Also header in the received INVITE(Also/Replace).*
To:	*Contents of the To: header MUST be filled in by the MTA as a locally unique string, as the identity of the call-ed party is not known to the MTA. MUST be different from the From: header.*
--all headers appearing in Dcs-Also: as additional headers--	*MUST be copied from the additional header component of the DCS-URL in the Dcs-Also header of the received INVITE(Also/Replace).*
--all other headers--	*All other headers are unchanged from Section 6.4.1*
	An empty line MUST be present between the headers and the message body
--SDP description--	*MUST be an SDP description, as specified in 6.4.1*

The CMS/Proxy receives the INVITE request, decrypts the information in the Request-URI, locates the CMS/Proxy associated with the new destination number, inserts Dcs-Billing-Info, Dcs-Billing-ID and Gate information, and forwards to the terminating CMS/Proxy.

On receiving an INVITE(also/replace) that includes the Dcs-Replaces: header, the MTA MUST verify it has a call active with the matching Call-ID and that the Dcs-Replaces value matches either the From or To call-leg identification. If all is proper, the MTA MUST send a BYE message to the party identified in the Dcs-Replaces header, and terminate that call.

6.8.7 Operator Services: Receipt of INVITE(BLV) and INVITE(EI)

Operator Services (Busy Line verification) and Emergency Interrupt are initiated from the PSTN, over special MF trunks groups from the OSPS system. The SIP messages INVITE(BLV) and INVITE(EI) are initiated by the PSTN gateway. These messages MUST include the Dcs-OSPS header. An INVITE(BLV) MUST have Dcs-OSPS set to BLV.

The MTA MUST be prepared to receive an INVITE(BLV) at any time. If not received from the CMS/Proxy, it SHOULD be rejected. It SHOULD NOT result in a busy error response. It MUST NOT result in alerting the user if the telephone is onhook. If the Dcs-Remoe-Party-ID header does not contain a rpi-type of "Operator," the MTA SHOULD reject the message.

Invite(BLV): (CMS/Proxy->MTA) Header:	***Requirement***
INVITE sip:E.164-t | host-t@Host(mta-t); user=phone | ip SIP/2.0	*MUST be present. MUST be sufficient to identify the line that is to be verified as busy.*
Dcs-Remote-Party-ID: User-o <tel:E.164-o>; rpi-type=Operator	*MUST be present. MUST contain Rpi-Type of "Operator."*
Dcs-Osps: BLV	*MUST be present. MUST be set to BLV.*
--all other headers, including SDP---	*MUST be as specified for a received INVITE (section 6.4.2)*

The MTA MUST respond to INVITE(BLV) with a 183-Session-Progress, and the call completes as in Section 6.4.2.1, 6.4.4, and 6.4.7.

The SDP describes the media flow from the MTA to the PSTN gateway; the MTA SHOULD send a packet stream to that address. The MTA MAY perform a mixing operation between the two ends of an active call, and send the mixed stream to the OSPS system. The MTA MAY check for voice activity locally, and if none send a copy of the received voice stream. The MTA MAY send a duplicate copy of the locally-generated voice stream.

If the telephone line is idle, the MTA SHOULD send a stream of silence packets to the OSPS system. If the telephone line is ringing, or locally generating a ringback tone, the MTA SHOULD send a ringback sequence to the OSPS system.

The operator may decide to interrupt the call after confirming that the line is busy, and signals this intention by placing an alerting tone on the voice path to the MTA. The PSTN Gateway detects this tone and formulates an INVITE(EI) message. This message is a variant of the INVITE with Dcs-OSPS header set to EI. This INVITE(EI) message is sent direct from the CMS/Agent of the PSTN Gateway to the MTA.

The MTA MUST be prepared to accept an INVITE(EI) at any time a BLV call is active. The INVITE(EI) is defined in the following table.

INVITE(EI): (EP -> MTA) Header:	***Requirements on MTA for message checking***

INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the Contact header from the 183-Session-Progress response for the INVITE(BLV) call.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present.* *MUST contain the IP address or FQDN of the originating CMS/Agent.*
From:	*MUST be present. MUST be same as the INVITE(BLV).*
To:	
Call-ID:	
CSeq: n INVITE	*MUST be present.* *Call sequence number "n" MUST be one greater than previous request message. Method MUST be INVITE.*
Dcs-OSPS: EI	*MUST be present. MUST be equal to EI*

If the MTA receives INVITE(EI) but has not previously received INVITE(BLV) with identical call-leg identification, it MUST reject the message.

On acceptance of a valid INVITE(EI), the MTA MUST respond with 200-OK, and enable communication between the operator and the local user. The MTA MAY place the existing call on hold and switch to the operator call (e.g. call-waiting). Alternatively, if resources are available, the MTA could establish a three-way call with the operator and the current party or parties.

6.8.8 SIP Messages for CODEC Changes – INVITE(Codec-change)

The INVITE(Codec-change) message is sent by either endpoint MTA to initiate a change in the codec. There are two separate cases described. First is a change to a codec that was in the original set of codecs listed in the initial INVITE request. Resource authorization has already been performed, and the Media-Authorization token received in the INVITE/183-Session-Progress can be used directly to increase the resources. The message exchange occurs only between the endpoints to synchronize the change.

The second case is a change to a coded that was not previously specified in the initial INVITE. The CMS/Proxies need to be involved in this to increase the resource authorization, and therefore the message exchange goes along the proxy path.

6.8.8.1 Codec Change within previous authorization

If the desired new codec was included in the SDP of the initial INVITE transaction (or authorized by a subsequent INVITE(Codec-Change) request), the codec is considered authorized by the network.

In this case, the MTA initiating the codec change MUST send an INVITE message directly to the other endpoint with the new codec description. To maintain privacy of the initiator, this INVITE message SHOULD NOT contain a Dcs-Remote-Party-ID header, nor a Dcs-Anonymity header. The format of the INVITE message sent by the initiating MTA (MTA_I) and the requirements on the header fields checked at the receiving MTA (MTA_R) are as follows.

INVITE(Codec-change): (MTA_I -> MTA_R) Header:	***Requirements on MTA_I for message generation Requirements on MTA_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the Contact header from the INVITE message or 183-Session-Progress response for this call.*
Via: SIP/2.0/UDP Host(mta-i)	*MUST be present.* *MUST contain the IP address or FQDN of the initiating MTA.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the change is initiated by the called party.*
To:	
Call-ID:	

CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 6.1.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *SHOULD include "a=X-pc-csuite" and "a=X-pc-secret" with the previous keying material, indicating no change is desired.* *MUST contain line "a=X-pc-qos: mandatory sendrecv"*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmission MUST stop on receipt of a final response.

On receiving an INVITE(Codec-change), MTA_R MUST match it to the existing call by the use of the From, To, and Call-ID headers. If there is no match, MTA_R considers this a new call attempt from a non-PacketCable endpoint, and MAY ignore it. MTA_R MUST send a 183-Session-Progress provisional response, giving the agreed codec.

183-Session-Progress: (MTA_R -> MTA_I) Header:	***Requirement for MTA_T for message generation*** ***Requirement for MTA_O for message checking***
SIP/2.0 183 Session Progress	*Status line with status code 183 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE message.* *Identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present. MUST be same as in 183-Session-Progress*
CSeq:	*MUST be present. MUST be the same as in the INVITE.*
RSeq: x_t	*MUST be present.*
Session: qos	*MUST be present, and MUST contain value "qos"*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *MUST contain "a=X-pc-qos: sendrecv mandatory confirm"*

The retransmission timer for this message SHOULD be set to T-direct-response. The default value of (T-direct-response) is given in Appendix A. Retransmissions MUST stop on receipt of PRACK.

MTA_I MUST send a PRACK to acknowledge receipt of the 183-Session-Progress. The PRACK message MUST be sent directly to the address specified in the Contact header.

An SDP MUST be included in the PRACK message. The SDP in the PRACK MUST include a media (m=) line with a single CODEC to be used for this connection.

PRACK: (MTA_I -> MTA_R) Header:	***Requirement at MTA for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the Contact header received in the initial 183-Session-Progress*
Via:	*MUST be present. MUST be the IP address or FQDN of MTA_O.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in the provisional response.*
Call-ID:	
Cseq: n_I+1 ACK	*Sequence number MUST be one higher than previous sequence number, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4.*
Content-length: (...)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after processing the SDP returned by the terminating MTA, and MUST contain a single CODEC choice.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

MTA_I MUST reserve the resources required and send a PRECONDITION-MET message, or other failure message, to MTA_R. This is as shown in 6.4.3.

MTA_R MUST use the SDP description in the PRACK message to reserve access network resources, and, if successful and after receiving a PRECONDITION-MET message from MTA_I, sends a 200-OK final response to MTA_I.

On sending the 200-OK, MTA_R commits network resources. It MAY start sending using the new codec.

200-OK: (MTA_R -> MTA_I) Header:	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the received INVITE.*
To:	
Call-ID:	*Identifies the call.*
CSeq:	*MUST be present. MUST be the same as in the INVITE.*

On receipt of a 200-OK response, MTA_I commits network resources and starts using the new codec. MTA_I sends an ACK to MTA_R. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

ACK: (MTA_I -> MTA_R) Header:	***Requirement at MTA_I for message generation***

ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the Contact header in the Initial INVITE or initial 183-Session-Progress.*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE(CodecChange) request.*
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(CodecChange) request, method MUST indicate ACK*

Example call flows for CODEC change within previous authorization are included in Appendix R.

6.8.8.2 Codec Change requiring new authorization

If the new codec desired by MTA_I was not included in the SDP of the initial INVITE and was therefore not authorized by the network, the MTA MUST send the INVITE(codec-change) request to its proxy. This message MUST have the same To:, From and Call-ID headers that identify the active call. The message traverses proxies like an INVITE message.

The format of the INVITE message sent by the initiating MTA (MTA_I) and the requirements on the header fields checked at the receiving MTA (MTA_R) are as follows.

INVITE(Codec-change): (MTA_I->CMS/Proxy->MTA_R) Header:	***Requirements on MTA_I for message generation Requirements on CMS/Proxy & MTA_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the Contact header from the INVITE message or 183-Session-Progress response for this call.*
Via: SIP/2.0/UDP Host(mta-I)	*MUST be present.* *MUST contain the IP address or FQDN of the originating MTA.*
Dcs-State:	*MUST be present in message sent from MTA_I to $CMS/Proxy_I$, and MUST contain value(s) given MTA_I by $CMS/Proxy_I$.*
From:	*MUST be present. MUST be same as initial INVITE for the call being modified, but with From: and To: reversed if the change is initiated by the called party.*
To:	
Call-ID:	
cSeq: I_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 6.1.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *SHOULD include "a=X-pc-csuite" and "a=X-pc-secret" with the previous keying material, indicating no change is desired.* *MUST contain line "a=X-pc-qos: mandatory sendrecv"*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmission MUST stop on receipt of a final response.

The remainder of the procedure for changing CODECs is identical to that described in Section 6.8.8.1, for a mid-call CODEC change that did not require an authorization change, except that MTA_R, as directed by the Via headers, sends its 183-Session-Progress and 200-OK responses to its proxy, and includes the Dcs-State header from its proxy in those responses. Both MTA_I and MTA_R MAY receive new Media-Authorization

tokens from the CMS/Proxies, and if so, MUST use these new Media-Authorization tokens in requesting the resources for the new CODEC.

An example call flow for a CODEC change requiring new authorization is included in Appendix S.

6.9 SIP Messages for Call Teardown

To terminate a call, the MTA MUST send a BYE message and stop transmitting bearer data to the other endpoint. It MUST release network resources used for the call.

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A.

We denote the MTA that has detected local hangup by MTA_I; the other MTA in the call is MTA_R;

BYE: (MTA_I -> MTA_R) **Header:**	***Requirement***
BYE sip:Host(mta-r) SIP/2.0	*Request line MUST include the BYE Method followed by the Contact header value of the destination*
Via: Host(mta-I)	*MUST be present*
From:	*From, To, Call ID MUST be present to identify the call leg to be torn down. This is identical to the initial INVITE for this call, except that the From and To may be reversed if the termination is requested by the called party.*
To:	
Call-ID:	
Cseq: n_I BYE	*The Sequence number MUST be as defined in 6.1.*

Upon receipt of the BYE message, MTA_R MUST release network resources that have been used for this call. MTA_R sends the following 200-OK message to MTA_I.

200-OK: (MTA_R -> MTA_I) **Header:**	***Requirement***
SIP/2.0 200 OK	*Status line MUST include status code 200.*
Via:	*MUST be present. MUST be coped from the BYE request.*
From:	*From, To, Call-ID MUST be present and MUST be copied from the preceding BYE request.*
To:	
Call-ID:	
CSeq:	*MUST be present. Same as in the preceding BYE.*

Upon receipt of 200-OK, MTA_I MUST stop the retransmission timer

6.10 MTA enabling Call Forwarding at the CMS/Proxy

If the MTA is unavailable, call forwarding can be enabled at the CMS/Proxy. The SIP Register method is used by the MTA to inform its CMS/Proxy of the number to which to forward. The CMS/Proxy MUST retain the single most recent REGISTER for each E.164 originator.

The message and the requirements on the headers are shown in the table below:

REGISTER: (MTA->CMS/Proxy) **Header:**	***Requirement at MTA for message generation*** ***Requirement at CMS/Proxy for message checking***
REGISTER sip:Host(dp-o) SIP/2.0	*MUST be present. Must be the address of the CMS/Proxy.*
Via: SIP/2.0/UDP Host(mta-o)	*MUST be present* *MUST be IP address or FQDN of MTA.*

From: sip:E.164-o@Host(dp-o); user=phone or From: tel:E.164-o	*MUST be present.* *Must include the phone number of the line being registered for the forwarding feature. If a SIP-URL, User=phone MUST be present.* *MAY be a tel: URL*
To: sip:Host(dp-o)	*MUST be present.* *MUST be the address of the CMS/Proxy*
Call-ID: ID	*MUST be present. MUST use this ID for all registrations and changes within this boot cycle.*
CSeq: n_o REGISTER	*MUST be present. Method MUST be Register.*
Contact: SIP-URL	*MUST include the address to which the call is to be forwarded. If the forwarding address is a phone number, either a tel: URL or a SIP-URL with user=phone MUST be present.*
Expires:	*MUST include non-zero value for the duration in seconds for which the registration is valid, or zero to remove a registration, or a SIP-date as defined in [11].*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. The retransmission timer is reset on receipt of a final response from the proxy.

An MTA receiving the 200-OK in response to the REGISTER MUST consider the registration successful. To remove a registration, the MTA MUST send a REGISTER request with Expires value of zero.

The Proxy MUST verify the MTA has subscribed to Call Forwarding service, else reject the request. When the registration is completed, the Proxy MUST send a 200-OK response.

The network registration call flow for the MTA unavailable is shown in Appendix I of this document.

7. CMS to CMS Interfaces

The Call Management Server (CMS) is an architectural entity that performs those services necesasry to enable endpoints to establish IP telephony calls. The CMS is a complex of server functions which support call signaling, number translation, and feature support. In addition to processing signaling messages, the CMS provides functions for authentication, service and feature authorization, call routing, service-specific admission control, as well as feature support for unavailable endpoints. As a trusted decision point, the CMS may also coordinate with Gate Controllers (which act as Policy Decision Points from a resource management point of view) to control when resource reservations are authorized for particular users.

This section describes the messages required to support IP Telephony between CMSs that support one or more of the following:

- Endpoints implementing Distributed Call Signaling (DCS), i.e. a DCS-Proxy (DP)
- Endpoints implementing Network-Based Call Signaling (NCS), i.e. a Call Agent (CA)
- PSTN Gateway Call Signaling (TGCP), i.e. a Media Gateway Controller (MGC)
- Various media servers, such as Announcement Servers, Bridge Servers, or VoiceMail Servers.
- Routing functions only, i.e. a tandem server within a service provider's domain, or a gateway server between service provider domains.

Section 6 provides additional detail on the Distributed Call Signaling interfaces from an MTA. Please refer to [8] for details on Network-Based Call Signaling, and to [9] for details on PSTN Gateway Call Signaling.

All of the various types of endpoint management systems fall into one of two different categories of CMS. A CMS/Proxy is the trusted entity that establishes calls on behalf of a SIP-enabled untrusted endpoint, e.g. an MTA in the customer premises, where the endpoint participates in the signaling exchanges directly. The role of the CMS/Proxy is to verify the signaling messages from the untrusted source, and provide various network services, such as translation, authentication and accounting. An example of a CMS/Proxy is a DCS-Proxy, as described in section 6 of this specification.

A CMS/Agent establishes connections either on its own behalf, or on behalf of a non-SIP endpoint. An example of the former is a voicemail server, and a conference bridge server; in these cases the CMS/Agent is a trusted network entity that establishes connections on its own behalf. Examples of the latter are the Call Agent (CA) described in NCS [8], the Media-Gateway-Controller (MGC) described in [9], and the Announcement Controller (ANC); in all of these cases there is another non-SIP protocol (NCS) exchanged between the CMS/Agent and the endpoint device, and the endpoint device does not participate in the SIP signaling exchanges directly. A combination of an MTA and its CMS/Proxy is in many ways equivalent to a CMS/Agent; likewise a CMS/Agent may be decomposed into an MTA and a proxy (with a hidden and untestable interface between them).

The term CMS in this specification refers to either of the above categories. Where only one category is being described, the term CMS/Proxy, or CMS/Agent will be used.

This section details the requirements on the syntax for messages that are sent or received by CMSs. This section also lists behavior requirements for CMSs when processing these messages. Unless otherwise stated in this text, CMS/Proxies MUST follow the same requirements given for SIP stateful Proxy Servers and

Registrars in RFC2543[11] Section 12, and CMS/Agents MUST follow the same requirements given for SIP user agents in RFC2543 Section 11.

7.1 Overview of CMS Behavior

PacketCable defines the Call Management Server (CMS) as a complex of server functions which support call signaling, number translation, call routing, feature support and admission control. Within the CMS complex, the PacketCable architecture allocates many of these responsibilities to the DP/CA/MGC and the Gate Controller (GC) function. In addition to processing call signaling messages, a CMS provides functions for authentication, service and feature authorization, name/number translation, call routing, service-specific admission control, as well as feature support for unavailable endpoints. As a trusted decision point, the CMS may also coordinate with Gate Controllers (which act as Policy Decision Points from a resource management point of view) to control when resources reservations are authorized for particular users. While the CMS is responsible for session control functions associated with proxying signaling requests, the Gate Controller is responsible for the policy decision regarding whether a requested QoS level should be admitted. Upon receipt of signaling information, a CMS instructs the Gate Controller to authorize a QoS level in advance of any resource management signaling.

The CMS associated with the endpoint originating a call is referred to as the originating CMS and is denoted as CMS_O. The CMS associated with the terminating endpoint is referred to as the terminating CMS and is denoted by CMS_T. The Gate Controllers (GC_O, GC_T) are the trusted policy decision points for controlling when and which resources are allowed to be reserved by users; they coordinate with the CMTSs ($CMTS_O$, $CMTS_T$) through D-QoS signaling. The CMTSs are the policy enforcement points, and ensure that the media path is provided the QoS it is authorized to receive.

The PacketCable CMS-CMS architecture extends the use of the basic INVITE/200-OK/ACK SIP transaction. A provisional response, the 183-Session-Progress, and its acknowledgement, the PRACK/200-OK, are used with the initial INVITE to exchange capabilities and establish call state in the network prior to alerting the user. Following this exchange, the endpoints engage in reservations to obtain the resources they will need for the media streams. If the resource reservations are successful, the originating endpoint performs a PRECONDITION-MET/200-OK exchange. At this point the initial INVITE continues with (possibly) a 180-Ringing, followed by the final 200-OK response and ACK. In all cases, all provisional and final responses to an INVITE message traverse the path taken by the original INVITE through one or more CMSs.

Variations of the basic INVITE, namely INVITE(replace), INVITE(also, replace), and INVITE(also), are used to modify the participants associated with a call, these messages are processed by the CMS so that appropriate reservation and billing modifications can be made and transferred to network entities such as CMTSs and Record Keeping Servers (RKS). INVITE(also) and INVITE(also, replace) are used to implement call features such as transfer and three-way-calling.

In support of billing functions, the originating CMS (CMS_O) includes information containing the account number of the caller and the Billing ID in the INVITE message that it sends/forwards. In support of resource management, the originating CMS also includes the location of the Gate.

Operator services such as Busy line verification (INVITE(BLV)) and Emergency interrupt (INVITE(EI)) are initiated from a Media-Gateway-Controller type of CMS/Agent and sent to the number being verified/interrupted.

7.1.1 CMS-CMS Interfaces

The interface between CMSs consists of two types of signaling messages, referred to as proxy-proxy signaling and end-end signaling. Proxy-proxy signaling consists of those messages involving call

authorization, billing, and various provider-controlled services. Any information considered confidential (to either the service provider or to the subscriber) is transported in the proxy-proxy signaling messages, and are never seen outside the trusted boundary of the service provider's network. End-end signaling consists of those messages performing synchronization and subscriber-controlled services.

Proxy-proxy signaling consists of the initial INVITE message used to establish the call, and, for a successful call, the 183-Session-Progress(qos) and 180-Ringing provisional responses, and the 200-OK final response. Any error responses to the initial INVITE are also carried on the proxy-proxy path, as are any redirection requests from the destination endpoint.

End-end signaling for a normal call setup consists of the provisional response acknowledgements (PRACK messages), and the PRECONDITION-MET message that indicates resources are available. The call termination (BYE message) is also done over the end-end signaling path.

Mid-call modifications to a call can be either proxy-proxy or end-end, depending on the nature of the change. Any change that involved an increase in the resources needed for media beyond those in the initial INVITE is sent on the proxy-proxy path. Any change in the call endpoints, such as call transfer, is sent on the proxy-proxy path. Changes such as call-hold and call-resume, and codec changes to lower bandwidth codecs are sent on the end-end path.

The proxy-proxy signaling also establishes a synchronization path that may be required by the Dynamic Quality of Service (D-QoS) specification [4] to coordinate the release of resources of the call. As per the D-QoS specification, the CMTS monitors the packet flow, and generates a Gate-Close message in response to either an explicit close request from the MTA/RGW, or when an equipment or facility failure causes the connection to be broken. This Gate-Close message is directed either to the local CMS/Agent or to the remote CMS/Agent or to the CMTS serving the remote MTA, depending on the capabilities of the endpoints. When a CMS/Agent receives such a Gate-Close message, it considers it identical to a call termination request.

Figure 8 below shows the basic set of configurations, and the particular associations maintained in the CMS-CMS interface for each.

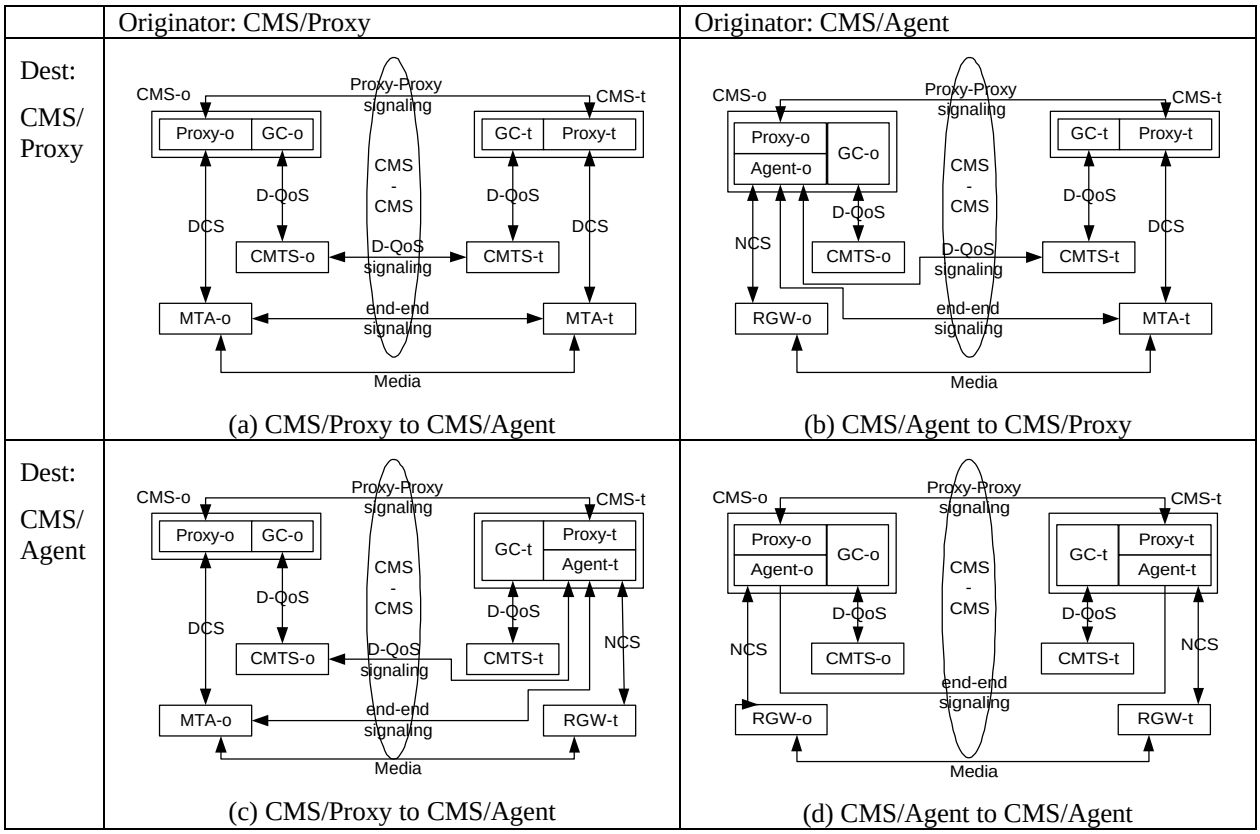


Figure 8: CMS-CMS Interfaces

Note that in case (d) there is no need for the D-QoS gate coordination, as the end-end signaling is sufficient to recognize call terminations and properly release resources. The CMS/Agent in cases (b), and (c) directs the local Gate-Close message to itself and internally notes the local hangup initiation; it directs the remote Gate-Close message to itself and uses it to internally note the remote hangup initiation.

Numerous other interfaces exist to the CMS, which are not shown in the above figure. These include translation servers, local-number-portability databases, SS7 signaling interfaces, and anonymizer interfaces.

Throughout this specification, the notation EP_O denotes the originating endpoint that performs the end-end signaling, and EP_T denotes the terminating endpoint that performs the end-end signaling. Thus, EP_O will mean either MTA_O in configurations (a) and (c), and CMS_O in configurations (b) and (d).

7.1.2 Overview of CMS/Proxy Behavior

The CMS/Proxy is a stateful SIP proxy. It maintains transaction state during call set-up and then MAY transfer encrypted call state information to the endpoints and network entities. The stateless CMS/Proxy does not maintain state about active calls. If an endpoint wishes to change an active call's characteristics (e.g. invoke calling features like forwarding, call transfer etc.), it passes the encrypted call state information in an INVITE request to its CMS/Proxy.

The Call Management Server (CMS) complex includes the CMS/Proxy and Gate Controller functions. The CMS/Proxy participates in the DCS signaling; the Gate Controller participates in the D-QoS signaling (see [4]). Together, they control the coordination of the signaling for call setup and resource management.

Messages for setting up a new call or changing the attributes or participants of an active call (that need participation from the service provider) go through the CMS/Proxies. In Figure 8(a), the paths labeled "DCS" and "Proxy-Proxy" show the route taken by signaling messages from one MTA to the other through CMS/Proxies.

7.1.2.1 CMS/Proxy behavior in support of call originator

When a user goes off-hook and dials a telephone number, the originating MTA (MTA_O) collects the dialed digits and initiates an INVITE request to the "originating" CMS/Proxy ($CMS/Proxy_O$). $CMS/Proxy_O$ verifies that MTA_O is a valid subscriber of the telephony service and determines whether this subscriber is authorized to place this call. $CMS/Proxy_O$ then translates the dialed number into the address of an MTA (if served by the same CMS/Proxy) or a "terminating" or "next hop" CMS/Proxy. $CMS/Proxy_O$ also modifies the Request-URI as necessary (e.g. if Local Number Portability lookup resulted in the expansion of the E.164 address).

$CMS/Proxy_O$ augments the INVITE message that it forwards with opaque pieces of information which contain the account number of the caller and the Billing ID (in support of billing functions), and the location of the Gate (in support of resource management). In addition, $CMS/Proxy_O$ generates and stores the encrypted Dcs-State header locally. This information is saved and sent to endpoint MTA_O when a final response is returned to MTA_O. $CMS/Proxy_O$ uses its interface to the local gate controller to allocate a gate at $CMTS_O$, and passes that gate identification in the INVITE message to establish the gate coordination path.

On receipt of the response to the INVITE message, $CMS/Proxy_O$ signals its Gate Controller to send a GATE-SETUP message to the originating CMTS ($CMTS_O$) to indicate that it can admit a gate for the IP flow associated with this call. This is in advance of any resource management signaling, which enables CMS_O to be "call-stateless". $CMS/Proxy_O$ generates an encrypted Dcs-State header containing all the call state information needed to support mid-call changes. $CMS/Proxy_O$ forwards the final response, including the Dcs-State header, to MTA_O.

All types of INVITE requests may contain encrypted call-state and billing information that can only be processed by the MTA's CMS/Proxy. Therefore, all these INVITE requests are sent via the CMS/Proxies to provide feature authorization and to trace the path back to the caller for features such as Return Call or Call Trace.

Call redirection can occur at two different points in the session establishment procedures. Prior to receipt of the 183-Session-Progress (and therefore prior to resource reservation) CMS_O performs the call redirection without the knowledge of MTA_O, thereby maintaining the privacy of the addressed destination. Once resources have been reserved and the originating user hears ringback, call redirection is performed by MTA_O. The new destination is hidden, however, through the use of a private URL for the new call.

7.1.2.2 CMS/Proxy behavior in support of call destination

CMS_O forwards the augmented request to $CMS/Proxy_T$ where the dialed number is translated into the address of the terminating MTA (MTA_T). After removing the billing information, $CMS/Proxy_T$ generates an encrypted Dcs-State header, retrieves local gate information, and appends the headers to the request forwarded to MTA_T to notify it about the incoming call.

If the line identified by the incoming request is available, MTA_T sends a 183-Session-Progress message to $CMS/Proxy_T$. The 183-Session-Progress(qos) contains the subset of the capabilities in the INVITE

message that are acceptable to MTA_T. $CMS/Proxy_T$ signals its Gate Controller to send a GATE-SETUP message to the terminating CMTS ($CMTS_T$), conveying policy instructions allowing $CMTS_T$ to open a gate for the IP flow associated with this phone call. The GATE-SETUP message contains billing information containing the account number of the subscriber that will pay for the call. This is done in advance of any resource management signaling, which keeps the CMS/Proxy "call-stateless" in that it does not need to maintain any state after the INVITE transaction completes. $CMS/Proxy_T$ forwards the provisional response to CMS_O.

Further provisional responses (e.g. 180-Ringing) and the final response (e.g. 200-OK) are forwarded by $CMS/Proxy_T$ as a stateless proxy, using only the via header information.

Call forwarding (e.g. call-forward-unconditional, call-forward-busy, and call-forward-no-answer) are requested by MTA_T through a 302-Redirect response to the INVITE. $CMS/Proxy_T$ verifies MTA_T subscribes to call fowarding service, and augments the response to provide the additional billing information needed for the redirected call.

7.1.2.3 CMS/Proxy behavior in support of mid-call changes

CMS/Proxies may also receive INVITE messages from a client during an active call, e.g. INVITE(Replace), INVITE(Also,Replace), and INVITE(codec-change). These are recognized at the CMS/Proxy by the presence of the Dcs-State header; absence of the Dcs-State header indicates a new call to be established, and presence of the Dcs-State header indicates a modification to a call already in progress.

Operator services such as Busy line verification (INVITE(BLV)) and Emergency interrupt (INVITE(EI)) are initiated from a CMS/Agent controlling a PSTN Media Gateway. INVITE(BLV) is sent via the CMSs to the number being verified/interrupted in order for the MTA to authenticate the caller-type. INVITE(EI) is sent end-to-end bypassing the CMS/Proxies, so as to retain the current behavior and expectations of the operator services.

Changes in attributes of a call that do not require CMS/Proxy participation, changes that do not affect the QoS characteristics of a call, and changes that do not affect the resource reservation may be performed end-to-end between MTAs directly. In Figure 8(a), direct signaling between MTAs follows the path labeled "end-end".

7.1.3 Overview of CMS/Agent Behavior

The CMS/Agent is a trusted SIP User Agent Client (UAC) and User Agent Server (UAS). It maintains call state during the life of the call, and monitors the endpoint device for state changes that affect the continuation of the call. The interface between the CMS/Agent and the endpoint device is outside the scope of this specification, but the particular case of Network-based Call Signaling (NCS) is used for examples.

The CMS/Agent can be considered to be an optimized combination of an untrusted MTA and a CMS/Proxy in a single element. The proxy portion of the CMS/Agent participates in the proxy-proxy signaling path, and the agent portion of the CMS/Agent participates in the end-end signaling path. The interface between the agent and proxy within a CMS/Agent is unspecified, but is not likely to match that of section 6. In particular, the information-hiding performed by a CMS/Proxy (e.g. generation of private URLs, encrypting Via headers, etc) is not necessary.

The Call Management Server (CMS) complex includes the CMS/Agent and, if needed, Gate Controller functions. The CMS/Agent participates in the CMS-CMS signaling; the Gate Controller participates, if needed, in the D-QoS signaling (see [4]). Together, they control the coordination of the signaling for call setup and resource management.

D-QoS signaling, consisting of the Gate-Open and Gate-Close message exchanges, is simulated by the CMS/Agent and used as a secondary mechanism to detect a call termination request. The CMS/Agent always directs the local CMTS to send its gate coordination messages to the CMS/Agent, and treats them as equivalent to a hangup request. When interworking with a CMS/Proxy, coordination with the remote CMTS provides an indication that the remote endpoint has terminated the connection, in cases when the CMS/Proxy is unable to detect this.

Messages for setting up a new call, or changing the attributes or participants of an active call, are initiated by the CMS/Agent or directed to the CMS/Agent. In Figure 8(d) both the proxy-proxy signaling path and the end-end signaling path go directly between CMS/Agents, while (b) and (c) show the paths when interworking with a CMS/Proxy.

7.1.3.1 CMS/Agent Behavior in support of call originator

Through some mechanism outside the scope of this specification, the CMS/Agent becomes aware that the endpoint device desires to initiate a connection, and determines the destination address of that desired connection. This may be done through a NTFY message in NCS, where the RGW detected the offhook condition and collected a complete dialstring from a sequence of touchtone button pushes. This may be done through a NTFY message in TGCP, where the MG detected a trunk seizure and received the destination address through MF signaling. This may be done through an IP-IAM message from a SS7 signaling gateway. Or this may be done through any number of other mechanisms.

The CMS/Agent ($CMS/Agent_O$) translates the destination address, and then takes the role of a trusted SIP UAC and initiates an INVITE request to the "terminating" CMS, or CMS_T. Included in this INVITE request is the SDP definition of the desired media stream, the billing/accounting information, the endpoint identification, and the indication of privacy requested by the call originator.

On receipt of the response to this INVITE request, $CMS/Agent_O$ authorizes the resources needed for the media stream, initiates any resource negotiation needed, and informs the destination endpoint when the resources are reserved. In response to a provisional response indicating alerting (180-Ringing), $CMS/Agent_O$ causes the endpoint device to play a ringback tone. In response to a final response, $CMS/Agent_O$ completes the connection and enables the media flow.

7.1.3.2 CMS/Agent Behavior in support of call destination

CMS_O forwards an INVITE request to $CMS/Agent_T$, where the dialed number is translated into the address of the terminating endpoint. Through negotiation with the terminating endpoint, $CMS/Agent_T$ determines the media stream properties, and authorizes QoS resources needed. $CMS/Agent_T$ responds to the INVITE request with a provisional 183-Session-Progress message, giving the SDP, destination identity information, and billing information (if the destination is overriding that given by the call originator).

$CMS/Agent_T$ directs the endpoint to reserve the resources necessary for the media stream. On receipt of the PRECONDITION-MET message from the originating endpoint, $CMS/Agent_T$ alerts the destination user, then completes the connection when the destination user answers.

Call features such as call-forwarding-unconditional, call-forwarding-busy, call-waiting, and call-forward-no-answer are controlled and implemented by $CMS/Agent_T$ by generating the proper SIP responses as part of the basic call setup procedures. $CMS/Agent_T$, locally storing information about the previous call, implements features such as return-call and call-trace.

7.1.3.3 CMS/Agent Behavior in support of mid-call changes

$CMS/Agent_O$ and $CMS/Agent_T$ monitor the endpoint during the call, and respond to any mid-call changes requested by the endpoint. Examples of such mid-call changes are hold/resume, codec change, call transfer, three-way-calling, operator busy-line-verification, and emergency interrupt. $CMS/Agent_O$ and $CMS/Agent_T$ initiate and perform the signaling exchanges necessary to make any of these changes.

7.2 CMS Retransmission, Reliability, and Recovery Strategies

The CMS MUST implement a retransmission timer to recover from lost request and response messages. SIP [11] defines a scheme based on two timer values, T1 and T2, where the retransmission interval starts at T1 seconds, and is doubled, with each attempt (up to a limit of T2 seconds), with a maximum number of retransmissions. DCS compliant CMSs MUST allow the value of T1 to be dependent on the request message being sent, and SHOULD implement a retransmission strategy using exponential back-off with a random offset, and configurable initial and maximum retransmission timer values.

DCS compliant CMSs MUST implement an additional timer, called T3 in this specification. Requirements on the conditions for setting this timer, and actions on its expiration, are given in section 7.6. On expiration of this timer, the CMS aborts the current request and returns to a known idle state.

$CMS/Proxy_O$ sets T3 to T-setup on receipt of the first provisional response to an INVITE, and cancels T3 on receipt of a 183-Session-Progress response to the INVITE.

$CMS/Proxy_T$ sets T3 to T-resource on receipt of an INVITE request, and cancels T3 on receipt of 183-Session-Progress response.

$CMS/Agent_O$ sets T3 to T-setup on receipt of the first provisional response to an INVITE, resets T3 to T-ringback on receipt of a 180-Ringing or 183-Media provisional response, and cancels T3 on receipt of a final response.

$CMS/Agent_T$ sets T3 to T-resource on receipt of an INVITE request, resets T3 to T-ringing on sending a 180-Ringing or 183-Media, and cancels T3 upon receipt of the final ACK message.

Default values for all of these timers (T-setup, T-ringback, T-resource, and T-ringing) are given in Appendix A.

When the provisioned number of message retransmissions is exceeded for an INVITE without any responses, the CMS MUST try a different CMS address, if available. When a provisioned number (which may be infinite) of CMSs have been tried, the CMS MUST return an error response.

The CMS MAY return a single 100-Trying provisional response to an INVITE request, and MUST return a single 100-Trying provisional response if more than 200ms will elapse before a provisional or final response will be sent to the sender. Such a provisional response will indicate to the sender that the message was received by the CMS; the sender will stop retransmissions and start the T3 timer. A CMS/Proxy MUST NOT initiate more than a single 1xx provisional response to the sender in response to the INVITE request, although it MUST pass on to the sender other 1xx provisional responses. Note that retransmission of the original INVITE request MAY cause a retransmission of the single 100-Trying provisional response to the sender.

7.3 CMS to CMS Routing

The CMS translates the destination address/phone-number, and obtains the address of the CMS that serves the destination, or the CMS for the next hop toward the destination. This translation function MAY involve a local-number-portability (LNP) database query as part of determining the proper destination, and if so, the CMS MUST generate a new Request-URI with the results of the LNP query. The CMS SHOULD include the hostname of the CMS that serves the destination in the Request-URI to reduce translation overhead in tandem proxies.

If the CMS is unable, for whatever reason, to send the request directly to the destination CMS, it determines a tandem. Choice of a tandem may be based on static configuration information, provisioning information, query of a routing function, or other methods. The CMS MUST check that it does not generate a request to a host listed in the Via-sent-by, Via-received or Via-maddr parameters.

If a CMS receives a request with a Request-URI identifying a destination other than the CMS, the CMS considers itself a tandem and attempts to send the request to its intended destination. The CMS/Tandem adds itself to the list of Via headers, so that it will be informed of the final response and clears all transaction state. In forwarding the request, the CMS/Tandem MUST check that it does not send the request to a CMS already listed in the Via headers. The Request-URI of the forwarded request SHOULD remain unchanged.

If a CMS receives a request identifying itself as the intended destination, performs the translation, and determines it is not the CMS serving the destination, the CMS considers itself a tandem and attempts to send the request to its intended destination. The CMS/Tandem adds itself to the list of Via headers, so that it will be informed of the final response and clears all transaction state. In forwarding the request, the CMS/Tandem MUST check that it does not send the request to a CMS already listed in the Via headers. The Request-URI MUST be rewritten to address the desired destination.

7.4 CMS messages

The CMS-CMS messages in this section are shown to use symbolic names in place of the actual values for various properties of the header fields. The meaning of the symbolic names and example values are defined in the table below.

Property	Origination	Destination	Example
Name	User-o	User-t	John Doe
MTA Name	MTA-o	MTA-t	
MTA address	Host(MTA-o)	Host(MTA-t)	44.20.0.3 or mta-o.provider
MTA port number	Port(MTA-o)	Port(MTA-t)	1234
CMS address	Host(CMS-o)	Host(CMS-t)	192.136.26.6 or cms-o.provider
Anonymizer IP address	Host(Ann-o)	Host(Ann-t)	
Anonymizer Port number	Port(Ann-o)	Port(Ann-t)	
Telephone number	E.164-o	E.164-t, E.164-f	123-456-7890
CMTS address	Host(CMTS-o)	Host(CMTS-t)	20.20.10.8 or cmts-o.provider
CMTS port number	Port(CMTS-o)	Port(CMTS-t)	4321
CMTS Gate ID	GID-o	GID-t	01AB926
Gate Key	GK-o	GK-t	
Dcs state	DS-o	DS-t	Always held in MTAs in encrypted form.

			Defined in 3.3.6.
RKS address	Host(RKS-o)	Host(RKS-t)	20.20.10.15 or rks-o.provider
RKS port number	Port(RKS-o)	Port(RKS-t)	4321
Account number	AC-o	AC-t	4278-9865-8765-9000

Attribute	Notation	Comments
Call-ID	ID	Random string. When generated by a CMS/Agent, the suggested implementation is a base64 encoding of a SHA-1 or MD5 cryptographic hash of local provisioned parameters (e.g. phone number) combined with a timestamp and a sequence number. Within a single message, multiple occurences of "ID" MAY be the same string
Call Sequence Number	n_o	Random starting sequence number chosen by CMS_O for the initial INVITE request.
	n_o+1 n_o+2 n_o+3	Numeric value one (two, or three) greater than the Call-Sequence-Number value used in the initial INVITE request.
	n_i	Call sequence number value used in a mid-call request message. Each client has an independently managed call sequence number for each call instance at that client. If sent by CMS_O, this value is one greater than the most recent Call-Sequence-Number sent in a request for this call by CMS_O. If the first request sent by CMS_T, this value is a random starting sequence number chosen by CMS_T (n_t). If a subsequent request sent by CMS_T, this value is one greater than the most recent Call-Sequence-Number sent in a request for this call by CMS_T.
Provisional Response Sequence Number	x_t x_t+1	Sequence number used in requesting an acknowledgement to a provisional response. If the first request for PRACK, this value is a random starting sequence number. If a subsequent request for a PRACK, this value is one greater than the most recent provisional response sequence number sent.
Billing-ID	BID	Unique Billing ID. Suggested implementation is a string made up of CMS_O's IP address, timestamp, and a sequence number.

7.5 General Requirements for headers

The table below lists general syntax and processing requirements for SIP header extensions in SIP messages received or sent by CMSs. The table also lists any additional requirements or exceptions for

standard headers in SIP messages received or sent by CMSs. All other headers are processed by the CMS according to the requirements listed in RFC2543[11].

Header Name	***Requirements, Comments***
Request Line	*MUST be present in all requests.* *MUST conform to rules for DCS-URLs as stated in section 4.2, but MUST NOT have private-param in request line.*
Via	*MUST be present in all requests and responses* *MUST contain IP address or FQDN of every CMS along routing path* *Response always sent to the address in the next Via header.*
From	*MUST be present in all requests and responses* *If Dcs-Anonymity is FULL, URL, Name or IPAddr, the username in addr-spec MUST be a random string that ensures privacy of the caller, the hostname MUST be the non-identifying name "localhost", and the display-name MUST be omitted.*
To	*MUST be present in all requests and responses* *If Dcs-Anonymity is OFF, then the To: header SHOULD contain a tel: URL with the dialed digits.* *If Dcs-Anonymity is FULL or URL or NAME, then the username in addr-spec MUST be a random string that is different from the From: header, the hostname MUST be the non-identifying name "localhost", and the display-name MUST be omitted*
Call-ID	*MUST be present in all requests and responses* *MUST be a unique string. Call-ID is an ASCII encoding of a random number designed to be unique over a period of several months*
Cseq	*MUST be present in all requests and responses*
Contact	*MUST be provided in initial INVITE request sent between CMSs.* *MUST be provided by CMS_T to CMS_O in the initial 183-Session-Progress response to INVITE.* *In initial INVITE and 183-Session-Progress, MUST be a SIP-URL in either IPv4 or FQDN form as described in section 4.2, and MUST NOT contain a private-param. If Dcs-Anonymity: header is set to IPAddr or Full, the Contact: header MUST contain a SIP-URL of an Anonymizer.* *MUST be present in 3xx-Redirect response, and MUST contain either a SIP-URL (possibly with "user=phone") or a tel: URL.*
Dcs-Gate	*MUST be provided in initial INVITE request sent between CMSs.* *MAY be provided by CMS_T to CMS_O in 183-Session-Progress response to INVITE.*
DCS-State	*MAY be present in messages sent between CMSs.*
Dcs-Remote-Party-ID	*MUST be provided in initial INVITE request sent between CMSs.* *MUST be provided by CMS_T to CMS_O in first 1xx, 2xx, or 3xx (except 100) response to INVITE.*
Dcs-Anonymity	*MAY be provided in initial INVITE request sent between CMSs.* *MAY be provided by CMS_T to CMS_O in first non-100 response to INVITE.*
Dcs-Also	*MAY be sent between CMSs in INVITE requests.* *MUST be a DCS-URL, but MUST NOT use private-param (i.e., MUST NOT be encrypted with a CMS-private key in messages between CMSs.)* *MAY have additional headers appended for Dcs-Replaces, Call-ID, Dcs-Billing-ID, Dcs-Billing-Info, Dcs-laes, and Dcs-Redirect.*
Dcs-Replaces	*MAY be sent between CMSs in INVITE requests.* *MUST be a SIP-URL.*
Dcs-Billing-Info	*MUST be provided in initial INVITE request sent between CMSs.* *MAY be provided by CMS_T to CMS_O in 183-Session-Progress response to INVITE.*
Dcs-Billing-ID	*MUST be provided in initial INVITE request sent between CMSs.*
Dcs-OSPS	*MAY be present in INVITE messages*
Dcs-Laes	*MAY be present in Initial INVITE or initial 183-Session-Progress response*
Dcs-Redirect	*MAY be present in INVITE messages*

7.6 CMS Messages and Procedures for Basic Call Setup

The basic INVITE message sequence for a DCS call setup include the INVITE/183-Session-Progress/180-Ringing(optional)/200-OK/ACK exchange, a PRECONDITION-MET/200-OK exchange, and one or two PRACK/200-OK message exchanges. These are shown in Figure 9, and discussed in the following subsections.

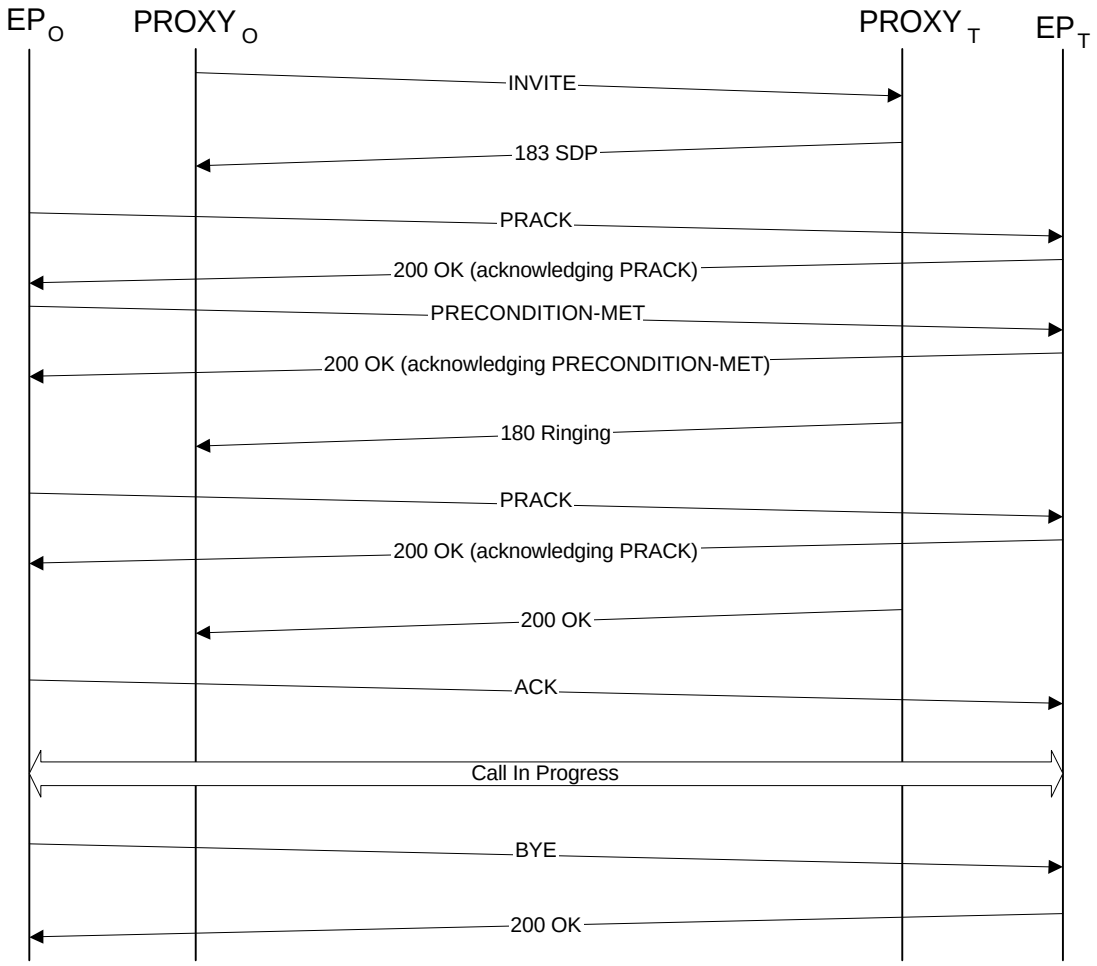


Figure 9: CMS Messages for Basic Call Setup

The INVITE message and the status responses to the INVITE are sent on the proxy-proxy path. The PRACK messages, the PRECONDITION-MET message, and the ACK to the INVITE's final status response is sent directly end-to-end between the caller and the callee.

The following sections trace a basic call from origination to completion, and give the requirements for each message exchange. It therefore switches viewpoints, from origination to termination, and back. For procedures followed by CMS_O initiating a call, see sections 7.6.1, 7.6.3, 7.6.6, and 7.6.8. For procedures followed by CMS_T in terminating a call, see sections 7.6.2, 7.6.4, 7.6.5, and 7.6.7. A typical CMS/Agent implements the procedures in all of these subsections, while specialized CMS/Agents only implement the portions needed in their application.

The architecture extends the syntax of the INVITE message SDP body with an attribute that permits caller and callee to exchange capabilities and to reserve necessary network resources prior to alerting the user.

The initial exchange consists of INVITE followed by 183-Session-Progress and a provisional acknowledgement (PRACK). Following this exchange, both endpoints know sufficient information to reserve the resources that will be needed to complete the call. Once those resources have been reserved, the call originator sends a PRECONDITION-MET message, and the destination continues the normal SIP processing with a 180-Ringing and/or 200-OK.

The behavior below also shows the procedures for call forwarding (unconditional and busy) and call forwarding (no answer).

If the CMS receives an INVITE message, the CMS MUST use its IPSec security association to determine the source of the message. If the source was an MTA, and the CMS is serving as a CMS/Proxy, then the CMS/Proxy is the originating proxy (CMS_O), and the procedures of 7.6.1 and 7.6.3 are followed. If the source was another CMS, the CMS is the terminating (CMS_T) or tandem (CMS/Tandem), and the procedures of section 7.6.2 are followed. Otherwise, the message is considered to be a standard SIP client attempting to contact a PacketCable endpoint, and MAY be ignored.

7.6.1 CMS_O initiating Invite

$CMS/Proxy_O$ becomes aware of a call origination attempt when it receives the INVITE message from MTA_O. A Call Agent (CA) becomes aware of a call origination attempt when it receives a NTFY message from the embedded client. A Media Gateway Controller (MGC) becomes aware of a call origination attempt when it receives a NTFY message from the media gateway, or a IP-IAM message from the signaling gateway.

CMS_O MUST use its IPSec security association to authenticate the source of the call origination request. $CMS/Agent_O$ MUST reject any request from a source other than a client being served. CMS_O MUST check that the indicated line is authorized for outgoing service. However, CMS_O MAY permit a call to an emergency service number even if originating service is not otherwise allowed.

The following call characteristics are determined by CMS_O, and used to generate the INVITE message.

- URL of the destination endpoint, either as a tel: url, or a sip: url. Typically this will be a phone number, and, if using the sip URL, will specify user=phone.
- Originating endpoint identification, both the originating phone number (or, in general, a URL of the originator), and the originating account name.
- The level of anonymity requested by the call originator
- Call leg identification, in the form of SIP From:, To:, and Call-ID: header values.
- Signaling address for end-end signaling messages.
- Session Description (SDP) for the media flow to the originating endpoint. This SDP includes all the required fields as given in Section 5, and all the choices of codecs (with appropriate rtpmap and bandwidth parameters).

This information is used to generate SIP headers as follows:

Header:	Requirements for $CMS/Proxy_O$	Requirements for $CMS/Agent_O$
Request URI	MUST be based on Request URI furnished by the MTA, as described in section 7.6.1.2 below	MUST conform to the rules for DCS-URLs as given in 4.2.
Dcs-Remote-Party-ID	MUST be based on the endpoint identification furnished by MTA, as described in section 7.6.1.1 below	Display-name MUST be one of a set of preprovisioned names allowed for the calling party. URL MUST contain the full E.164 phone number of the calling party, either as a tel: URL or as a DCS-URL with telephon-subscriber syntax and user=phone.
Dcs-Anonymity	MUST be unchanged from that given by the MTA, except as provided in section 7.6.1.2 below.	MUST be either OFF, FULL, URL, NAME, or IPADDR, or a valid combination as given in 3.3.3. If the caller has not requested privacy, MUST be OFF. If the caller has requested privacy, MUST be FULL, or a combination

		of URL, NAME, and IPADDR
From	MUST be unchanged from that given by MTA	MUST follow the requirements of section 4.6.6. It MAY identify the caller by name, IP address, or by phone number If Dcs-Anonymity is FULL, URL, Name or IPAddr, the username in addr-spec MUST be a random string that ensures privacy of the caller, the hostname MUST be the non-identifying name "localhost", and the display-name MUST be omitted.
To	MUST be unchanged from that given by MTA	MUST follow the requirements of section 4.6.7. The To: header MAY contain the URI of the callee If Dcs-Anonymity is OFF, then the To: header SHOULD contain a tel: URI with the dialed digits. If Dcs-Anonymity is FULL or URL or NAME, then the username in addr-spec MUST be a random string that is different from the From: header, the hostname MUST be the non-identifying name "localhost", and the display-name MUST be omitted.
Call-ID	MUST be unchanged from that given by MTA	MUST be a unique string. Call-ID is an ASCII encoding of a random number designed to be unique over a period of serveral months.
Contact	MUST be unchanged from that given by MTA, except as modified by 7.6.1.4	SHOULD be Host(cms-o), or Host(ann-o) if Dcs-Anonymity is Full or IPAddr.
SDP	MUST be based on SDP furnished by MTA, as described below	The session description MUST include the list of CODECs the originating endpoint is willing to support for this connection. It MUST include on the media (m=) line the preferred CODEC for which resources MUST be available in the endpoint, and available to receive and play payload packets. It MAY include an attribute (a=X-pc-codecs) line giving alternatives available. The SDP MAY be generated entirely by the CMS/Agent, or MAY be generated by interactions between the CMS/Agent and the endpoint beyond the scope of this specification. The SDP sent CMS-CMS MUST include a qos precondition of the form "a=X-pc-qos:mandatory"

7.6.1.1 CMS/Proxy$_O$ Authentication and Authorization of Originator

If a Dcs-Remote-Party-ID header is present, CMS/Proxy$_O$ MUST check that the originating phone number belongs to MTA$_O$, and that the display name (if present) is a valid calling name for this line.

If the Dcs-Remote-Party-ID header is not present, or if the Dcs-Remote-Party-ID is present and incorrect, the CMS/Proxy$_O$ MUST generate a default Dcs-Remote-Party-ID header value based on provisioned information. If no provisioned information is available (e.g. if the MTA serves a multiple-dwelling-unit), CMS/Proxy$_O$ MUST reject the call attempt. However, CMS/Proxy$_O$ MAY permit a call to an emergency service number even if the Dcs-Remote-Party-ID header is not present or invalid.

If the destination address contains any request for a special service, or if the username of the Request-URI contans the special service name "bridge" or "call-trace", the CMS/Proxy MUST verify that the calling line is authorized for the requested service. If the line is not authorized for the service requested, CMS/Proxy$_O$ MUST reject the call attempt with an appropriate error.

7.6.1.2 CMS/Proxy$_O$ Initial Processing of Request URI

If the Request-URI contains the private-param user parameter, the CMS/Proxy MUST decrypt the username information to find the real destination for the call, and other special processing information. If electronic surveillance information is contained in the decrypted username, the CMS MUST generate a Dcs-LAES header with the surveillance information. If billing information is contained in the decrypted

username, the CMS MUST generate a Dcs-Billing-Info header with the billing information. If a privacy indication is contained in the decrypted username, then the CMS MUST merge this privacy request with the Dcs-Anonymity header. The intended recipient's SIP-URL MUST be derived from the decrypted contents of the username, and any special-service-name in the Request-URI.

If the username cannot be decrypted, or the checksum contained in the decrypted username is incorrect, or the signature contained in the decrypted username is incorrect, or the expiration time contained within the decrypted username is considered invalid or too old, the call MUST be rejected and an appropriate 4xx, 5xx, or 6xx response MUST be returned to MTA_O.

Dcs-Also headers, if present, MUST be handled as described in section 7.8.1.2.1.

7.6.1.3 Address Translation

CMS_O MUST resolve the destination number from the Request-URI into 1) the address of a destination endpoint served by this CMS, 2) the address of another CMS, or 3) consider the request to be in error.

If CMS_O performs the local-number-portability lookup, it MUST generate a Request-URI containing a DCS-URL with the user-param "user=np-queried." CMS_O MUST include the lrn-tag indicating the returned value if the local-number-portability lookup returned a value.

7.6.1.4 Anonymizing support.

CMS_O checks for user-requested privacy by examining the Dcs-Anonymity header line.

If Dcs-Anonymity is set to "Full" or "IPADDR", then CMS_O MUST provide IP address privacy through the use of an anonymizer service. CMS_O MUST allocate a port on the anonymizer, using the protocol specified in section 8.4. CMS_O MUST modify the SDP "c=" line, and modify the "Contact:" header to be the address and port at the anonymizer.

See Appendix Z for an example.

7.6.1.5 INVITE message generation

If the destination endpoint is not served by this CMS, CMS_O generates a SIP INVITE message and sends it to CMS_T, the CMS that manages the terminating endpoint.

CMS_O MUST add its IP address or FQDN to the top of the Via header list, and MUST add a "branch=x" parameter, where x is a unique number for this transaction.

CMS_O MUST add the Dcs-Billing-Info and Dcs-Billing-ID headers, which are defined in 3.3.9. The semantics of Dcs-Billing-Info and Dcs-Billing-ID are described in [3].

CMS_O MUST add the Dcs-Gate header to support D-QoS gate coordination. The use of gate coordination is described in [4]. $CMS/Proxy_O$ SHOULD give the address of the $CMTS_O$, the Gate-ID within $CMTS_O$, and the security key and cipher suite expected by $CMTS_O$ for gate coordination messages related to that gate. $CMS/Agent_O$ SHOULD give its address, port, security key, and cipher suite for handling simulated gate coordination messages. $CMS/Agent_O$ SHOULD omit the gate-strength token, or specify the gate-strength-token as "optional"; $CMS/Proxy_O$ SHOULD specify the gate-strength-token as "required".

CMS_O SHOULD add a Dcs-State header, including state information it needs to process mid-call signaling messages that originate at the called party. In order to achieve stateless operation, a CMS/Proxy SHOULD include in this Dcs-State header the Gate location and identifier (for retrieving the billing identifier and

accounting information), and the original destination and redirection count (in support of Electronic Surveillance).

CMS_O MUST insert a security key into a a=X-pc-secret line of the SDP. See [2].

INVITE (CMS_O -> CMS_T) Header:	***Additional requirements for message generation***
INVITE DCS-URL SIP/2.0	*As described above*
Via: SIP/2.0/UDP Host(cms-o);branch=x	*MUST be present.* *MUST contain the IP address or FQDN of CMS_O.* *MUST contain branch=x, where x is a unque number for this transaction.*
Via: SIP/2.0/UDP Host(mta-o)	*CMS/Proxy only: all Vias MUST be copied from INVITE received from MTA*
Supported: org.ietf.sip.100rel	*MUST be present. MUST indicate org.ietf.sip.100rel*
Dcs-Remote-Party-ID:	*As described above*
Dcs-Anonymity:	*As described above*
Dcs-Gate:	*As described above*
Dcs-Billing-ID:	*As described above*
Dcs-Billing-Info:	*As described above*
Dcs-State:	*SHOULD be present, as described above.*
From:	*As described above*
To:	*As described above*
Call-ID:	*As described above*
CSeq: n_o INVITE	*MUST be present.* *Call sequence number "n_o" and the request method MUST be present.* *CMS/Proxy only: MUST be copied from INVITE received from MTA*
Contact:	*As described above. MAY be modified in support of IP address privacy*
---all other headers provided by MTA to CMS/Proxy---	*CMS/Proxy only: MUST be copied from INVITE received from MTA*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*As described above* *c= line MAY be modified in support of IP address privacy* *CMS_O MUST add a "a=X-pc-secret" line to the SDP, giving a security key to be used by the media packets for this session (e.g. clear:RC4/ItWasTheBestOfTimesItWasTheWorstOfTimes), unless one was already present.*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of any response.

CMS_O MUST accept a 100-Trying message as described in the following table.

100-Trying (CMS_T -> CMS_O) Header :	***Requirement at CMS for message checking***
SIP/2.0 100 Trying	*Status line with status code 100 MUST be present.*
Via:	*MUST be same as transmitted INVITE message.*
From:	*From, To, CallID and Cseq MUST be same as the transmitted INVITE message, except that a tag-param MAY be added, as per RFC2543 [11].*
To:	
Call-ID:	
CSeq:	

On receipt of a 100-Trying provisional response, the retransmission timer T1 MUST be cancelled.

For CMS/$Agent_O$, the transaction timer (T3) for this exchange MUST be set to T-setup. The default value of (T-setup) is given in Appendix A. On expiration of T3, CMS/$Agent_O$ MUST clear the call attempt, and

send a CANCEL message to CMS_T with the same values of Request-URI, From, To, and CallID, and any Dcs-State headers received for this call attempt.

For CMS/$Proxy_O$, the transaction timer (T3) for this exchange MUST be set to T-proxy-setup. The default value of (T-proxy-setup) is given in Appendix A. The T3 timer is cancelled on receipt of a 183-Session-Progress, a final response, or a CANCEL or BYE request. On expiration of T3, CMS/$Proxy_O$ MUST return a 480-Temporarily-Unavailable error response to MTA_O.

7.6.2 Invite from CMS_O arrives at CMS_T

Upon receiving an INVITE message, CMS_T MUST authenticate, by using IPSec, that the sender is properly identified in the topmost Via header.

CMS_T MUST resolve the destination number from the Request-URI into 1) the address of a destination endpoint served by this CMS, 2) the address of another CMS, or 3) consider the request to be in error.

If the results of the Request-URI translation is a different CMS, then this is a CMS/Tandem operation. The CMS/Tandem MUST add a Via header to the INVITE message identifying itself. The CMS/Tandem identifies the routing to deliver the INVITE to its intended destination, CMS_T. If the hostname of the Request-URI contained the CMS/Tandem address, CMS/Tandem SHOULD include a Dcs-State header in the INVITE message containing this routing information for the forward and/or reverse directions. If the Request-URI did not contain the CMS/Tandem address, CMS/Tandem SHOULD NOT include a DCS-State header in the INVITE message. The CMS/Tandem sends the INVITE message toward the proper CMS identified by translation (as described in section 7.3), and MAY act as a stateless proxy in the handling of all the provisional and final responses.

If the translation described above determined that the destination endpoint is served by this CMS, processing continues as specified in this section. CMS_T MUST determine the local endpoint being addressed by this call. CMS_T MUST check to see if this endpoint is authorized to receive this call. If translation or authorization fails, CMS_T MUST return an appropriate 4xx, 5xx, 6xx error code.

On receiving the INVITE message, CMS/$Agent_T$ MUST start the transaction timer (T3) with value T-resource. The default value of (T-resource) is given in Appendix A. Timer T3 is canceled by the completion of resource reservation and receipt of a PRECONDITION-MET message. On expiration of timer T3, CMS/$Agent_T$ MUST send a 480-Temporarily-Unavailable or 302-Redirect response to CMS_O.

On receiving the INVITE message, CMS/$Proxy_T$ MUST start the transaction timer (T3) with value T-proxy-setup. The default value of (T-proxy-setup) is given in Appendix A. Timer T3 is canceled by the receipt of a 183-Session-Progress message. On expiration of timer T3, CMS/$Proxy_T$ MUST send a 480-Temporarily-Unavailable or 302-Redirect response to CMS_O.

CMS_T determines, possibly by consultation with the endpoint, whether it will accept the call, or forward to another destination, or return an error (such as busy). The following call characteristics are determined by CMS_T, and used to generate the INVITE response.

- If endpoint will accept the call:
  - Destination endpoint identification, both the terminating phone number (or, in general, a URL of the destination), and the destination account name
  - The level of anonymity requested by the call destination.
  - Signaling address for end-end signaling messages
  - Session Description (SDP) for the media flow to the destination endpoint. This SDP includes all the required fields as given in Section 5, and a subset of the choices of codecs (with appropriate rtpmap and bandwidth parameters) that are acceptable to the destination endpoint.

- If endpoint will forward the call to another destination, the URL of the new destination
- If endpoint returns an error, the specific error code (such as busy)

The mechanism by which this is done for CMS/Agent$_T$ is outside the scope of this specification. The following defines the procedures for CMS/Proxy$_T$.

7.6.2.1 CMS/Proxy$_T$ Actions

CMS/Proxy$_T$ MUST generate an INVITE message for delivery to the destination MTA, MTA$_T$. Various fields are modified or deleted prior to delivery to MTA$_T$. This section gives the requirements for the header modifications, while section 6.4.2 gives the requirements for the INVITE message and the possible responses from MTA$_T$.

CMS/Proxy$_T$ MUST add a Via header containing the IP address or FQDN of CMS$_T$. The remaining Via headers MUST be stored for use later in handling of provisional and final responses. If Full or IP-Address anonymity is requested, then the remaining Via headers MUST be hidden from MTA$_T$ and encrypted. The encrypted information MAY be contained in the Dcs-State string with a CMS-defined token as the second component of the Via header, or MAY be passed to MTA$_T$ as an encrypted second component of the Via header.

CMS/Proxy$_T$ MUST generate a Dcs-State header information for MTA$_T$, sign and encrypt the data with its private key, and append the header to the INVITE that is formed for MTA$_T$. This state information SHOULD contain gate location and gate-id at CMTS$_T$, the Dcs-Laes and Dcs-Redirect values, if present, and a concatenation of all other Dcs-State headers from other proxies in the path. Additional information such as Billing-ID, and Billing-Info MAY be contained in the Dcs-State string, or MAY be obtained when needed from the gate parameters.

CMS/Proxy$_T$ MUST append the Dcs-Media-Authorization header containing information identifying the gate at the terminating CMTS$_T$.

If the caller has requested privacy with Dcs-Anonymity: Full or URL, CMS/Proxy$_T$ MUST replace the URL in the Dcs-Remote-Party-ID header with a private URL and add “rpi-id=private”. The private URL MUST be formed by encrypting the original URL and Dcs-Anonymity value (and possibly additional information) with CMS/Proxy$_T$’s privately held key, placing the resulting string as the username, inserting CMS/Proxy$_T$ as hostname, and adding a url-parameter of “private”. If the caller has requested privacy with Dcs-Anonymity: Full or Name, CMS/Proxy$_T$ MUST delete the display-name in the Dcs-Remote-Party-ID header. If the callee has not subscribed to calling name delivery, then CMS/Proxy$_T$ MUST delete the display-name in the Dcs-Remote-Party-ID header. If the callee has not subscribed to calling number delivery, then CMS/Proxy$_T$ MUST replace the URL in the Dcs-Remote-Party-ID header with a private URL (as described above) and add “rpi-id=na”. CMS/Proxy$_T$ MUST in all cases delete the Dcs-Anonymity header from the INVITE message.

CMS/Proxy$_T$ MUST NOT send the Dcs-Billing-ID, Dcs-Billing-Info, Dcs-Gate, Dcs-Laes, nor Dcs-Redirect headers to MTA$_T$.

CMS/Proxy$_T$ forwards this modified INVITE to MTA$_T$ and receives a response, as described in 6.4.2. Contents of the response determine which of the following subsections applies.

7.6.2.2 CMS$_T$ Sending 183-Session-Progress status response

If the destination endpoint is able to accept the call, CMS$_T$ forwards the 183-Session-Progress provisional response to CMS$_O$. The response’s session description MUST indicate the CODECs that the destination endpoint is willing to support, and MUST be a subset of those received in the INVITE.

CMS_T checks for an outstanding lawfully authorized surveillance order for the terminating subscriber. If present, CMS_T includes this information in the authorization for Quality of Service. If the terminating equipment is unable to perform the required surveillance (e.g. if the destination is a voicemail server), CMS_T MUST include a Dcs-Laes header in the 183-Session-Progress response to CMS_O requesting it to perform the surveillance. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be intercepted, and MUST include a random string for use as a security key between the Delivery Functions.

CMS_T uses the information in the SDP description, the electronic surveillance indication, and the Billing-ID and Billing-Info header values to signal the terminating Gate Controller (GC_T) to send the GATE-SETUP command to the terminating CMTS ($CMTS_T$), defining the envelope of the authorized QoS parameters.

CMS_T MUST check the Dcs-Anonymity header to see if the called party has requested IPADDR Privacy. If IPADDR Privacy has been requested, then CMS_T MUST provide IP address privacy through the use of an anonymizer service. CMS_T MUST modify the Contact and SDP connection information based on the IP addresses provided by the anonymizer service in the response that is sent to CMS_O. See Section 8 and Appendix Z for further details and examples.

If the INVITE request sent to CMS_T included the "mandatory" attribute on the Dcs-Gate header, or if CMS_T is a CMS/Proxy, CMS_T MUST add the Dcs-Gate header in its response. The CMS/Proxy SHOULD give the address of the $CMTS_T$, the Gate-ID within $CMTS_T$, and the security key and cipher suite expected by $CMTS_T$ for gate coordination messages related to that gate. The CMS/Agent, if including this header, SHOULD give its address, port, security key, and cipher suite for handling simulated gate coordination messages.

CMS_T MAY include a Dcs-Billing-Info header if it wishes to override the billing information that came in the INVITE (e.g. for a toll-free call).

$CMS/Proxy_T$ SHOULD add a Dcs-State header, including state information it needs to process mid-call signaling messages that originate at the called party. In order to achieve stateless operation, $CMS/Proxy_T$ SHOULD include in this Dcs-State header the Gate location and identifier (for retrieving the billing identifier and accounting information), and the original remote-party and redirection count (in support of Electronic Surveillance).

The 183-Session-Progress provisional response sent by CMS_T to CMS_O MUST be as follows:

183-Session-Progress (CMS_T -> CMS_O) Header :	***Requirement of CMS_T for message generation***
SIP/2.0 183 Session Progress	*Status line with status code 183 MUST be present.*
Via:	*MUST be copied from received INVITE message.*
Dcs-Remote-Party-ID: [User-t] <tel:E.164-t>	*MUST be present.* *MUST represent the same calling party as the Contact: header.* *URL MUST contain the phone number of the called party, either as a tel: URL, or as a SIP-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the called party.*
Dcs-Anonymity: OFF | FULL | URL | NAME | IPADDR	*If the Call-ee has requested privacy, this header MUST be present and MUST be FULL, URL, NAME, or IPADDR. If the callee has not requested privacy, this header MAY be present , and if present, MUST be OFF.*
Dcs-State: Host(cms-o); DS-o	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
Dcs-State: Host(cms-t); DS-t	*Additional Dcs-State header generated by CMS_T MAY be present*
Dcs-Gate:	*As described above*
Dcs-Billing-Info	*As described above*
From:	*From, To, CallID and CSeq MUST be copied from received INVITE message, and a tag-param MAY be added, as per RFC2543 [11].*

To:	
Call-ID:	
CSeq:	
Contact:	*MUST be inserted by CMS_T as the address for future end-end signaling messages for this call. MAY be the address of an Anonymizer.*
Session: qos	*MUST be present. MUST containe 'qos' and MAY also contain 'security'. MUST NOT contain 'media'*
Rseq: x_t	*MUST be present. MUST contain the initial random sequence number chosen (for a CMS/Proxy) by MTA_T, or by the CMS/Agent*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..* *The response to INVITE must contain the SDP description of the media stream to be sent to the destination endpoint*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *SDP description of media streams acceptable to the destination endpoint, as described in Section 5.* *MUST contain a line 'a=X-pc-qos:mandatory' with attribute 'confirm'*

For $CMS/Agent_T$, the retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of the matching PRACK.

For $CMS/Proxy_T$ there is no retransmission of this message, as the MTA performs the required retransmissions and receives the PRACK message. The CMS/Proxy MUST store the 183-Session-Progress response for a period of time (32 seconds[11]) to handle retransmissions of the 183-Session-Progress from MTA_T. At the end of this period, CMS_T MAY delete all its state information related to this call and process all remaining provisional and final responses as a stateless proxy.

$CMS/Proxy_T$ MUST cancel the session timer (T3) upon sending the 183-Session-Progress response.

7.6.2.3 CMS_T Sending 3xx REDIRECT status response

Procedures in this section are invoked when CMS_T determines (by methods of section 6 for a CMS/Proxy, or by methods beyond the scope of this specification for a CMS/Agent) that the incoming call is to be forwarded. CMS_T MUST verify that the called party is a subscriber to the Call Forwarding service. If not, CMS_T MUST send a 480 Temporarily Unavailable error response to CMS_O**.**

CMS_T MUST check for an outstanding lawfully authorized surveillance order for the terminating subscriber. If found, CMS_T MUST include a Dcs-LAES header in the 3xx-Redirect response. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be delivered, and MUST include a random string for use as a security key between the Delivery Functions.

CMS_T MUST add a Dcs-Billing-Info header to the response to allow the "second leg" of the forwarded call to be charged to the terminating party, and MUST copy the Dcs-Billing-Info and Dcs-Billing-ID provided by CMS_O in the request.

CMS_T MUST forward the following 3xx-Redirect response to CMS_O.

302-Redirect (CMS_T -> CMS_O) Header:	***Requirement on CMS_T for message generation Requirement on CMS_O for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
Dcs-Billing-ID:	*MUST be copied from the INVITE message*
Dcs-Billing-Info: Host(rks-o):Port(rks-o)<AC-o/E.164-o/E.164-t>	*MUST be copied from the INVITE message*
Dcs-Billing-Info: Host(rks-t):Port(rks-t)<AC-t/E.164-t/E.164-f>	*New Billing-Info header as described above*
Dcs-Laes:	*MAY be present, as described above*
Dcs-Remote-Party-ID: [User-t] <tel:E.164-t>	*MUST be present.* *URL MUST contain the phone number of the called party, either as a tel: URL, or as a SIP-URL with telephone-subscriber syntax and user=phone.* *Display-name MAY be present, and if present, MUST be one of a set of preprovisioned names allowed for the called party.*
Dcs-Anonymity: OFF | FULL | URL | NAME | IPADDR	*If the Call-ee has requested privacy, this header MUST be present and MUST be FULL, URL, NAME, or IPADDR. If the callee has not requested privacy, this header MAY be present , and if present, MUST be OFF.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be inserted by CMS_T and carries the new destination information. It MUST be a valid URI.* *If the new destination is a telephone number, then the format of the URI MUST be a tel: URL where the URL contains a full E.164 number.*
Expires:	*MAY be present*

CMS_T MUST now take responsibility for delivery of the 3xx response to CMS_O. The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A.

Retransmissions MUST stop on receipt of the following ACK message.

ACK : (CMS_O -> CMS_T) Header:	***Requirement at CMS_T for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack. Request-URI MUST be copy of initial INVITE.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O*
From:	*MUST be present.* *MUST be copies of same headers in 302 response from CMS_T.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE, method MUST indicate ACK*

7.6.2.4 CMS_T Sending Other Status Response to INVITE request

A final error response, 4xx, 5xx, or 6xx response, MUST be sent as per [11]. This includes, but is not limited to, 486-Busy Here. The error response MUST be generated as follows.

Error: (CMS_T -> CMS_O) Header:	***Requirement on CMS_T for message generation Requirement on CMS_O for message checking***

SIP/2.0 xxx	*Status line header MUST be presentt. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A.

Retransmissions MUST stop on receipt of the following ACK.

ACK: (CMS_O -> CMS_T) Header:	***Requirement at CMS_T for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack. Request-URI MUST be copy of initial INVITE.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present. MUST be copies of same headers in error response from CMS_T.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE, method MUST indicate ACK*

7.6.3 CMS_O Receives Initial status response

In response to the initial INVITE request, CMS_O MUST be prepared to receive a 183-Session-Progress provisional response (in a normal call establishment), a 3xx-Redirect response (if the call was forwarded), or a 4xx-5xx-6xx error response (error cases, such as busy). Final responses, including 4xx-5xx-6xx, are described in section 7.6.8. Other initial responses, such as 180-Ringing and 200-OK, would be generated by a non-DCS endpoint, and MAY be ignored.

7.6.3.1 CMS_O handling of 183-Session-Progress response

The 183-Session-Progress provisional response received by CMS_O MUST be checked as follows:

183-Session-Progress (CMS_T -> CMS_O) Header :	***Requirement of CMS_O for message checking***
SIP/2.0 183 Session Progress	*Status line with status code 183 MUST be present.*
Via:	*MUST be same as was sent in INVITE message to CMS_T.*
Dcs-Remote-Party-ID:	*MUST be present.*
Dcs-Anonymity:	*MAY be present.*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value. Additional Dcs-State header MAY be present from CMS_T*
Dcs-Gate:	*MAY be present.*
Dcs-Billing-Info:	*MAY be present.*
From:	*From, To, CallID and CSeq MUST be identical to transmitted INVITE message, and a tag-param MAY be added, as per RFC2543 [11].*
To:	
Call-ID:	
CSeq:	
Contact:	*MUST be present*
Session: qos	*MUST be present. MUST containe 'qos' and MAY also contain 'security'. MUST NOT contain 'media'*

Rseq: x_t	*MUST be present.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..* *The response to INVITE must contain the SDP description of the media stream to be sent to the destination endpoint*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *SDP description of media streams acceptable to the destination endpoint, as described in Section 5.* *MUST contain a line 'a=X-pc-qos:mandatory' with attribute 'confirm'*

The action taken on receipt of this provisional status message is either sending of a PRACK (if CMS/Agent) or passing the 183-Session-Progress to MTA_O (if CMS/Proxy). The common processing of this message is described first.

7.6.3.1.1 Electronic Surveillance Support

CMS_O checks for an outstanding lawfully authorized surveillance order for the originating subscriber, and, if present, includes this information in the Authorization for Quality of Service or signals this information to the device performing the intercept (e.g. a Media Gateway).

If the Dcs-LAES header is present in the 183-Session-Progress response (indicating surveillance is required on the terminating subscriber, but that the terminating equipment is unable to perform that function), CMS_O MUST include this information in the Authorization for Quality of Service, or MUST signal this information to the device performing the intercept (e.g. a Media Gateway).

7.6.3.1.2 Authorization for Quality of Service

CMS_O MUST use the list of codecs specified in the SDP payload to authorize maximum resources that can be used during this call at the originating CMTS ($CMTS_O$). This information is used to signal the originating Gate Controller to send the GATE-SETUP command to the originating CMTS, defining the envelope of the authorized QoS parameters. Also included in the GATE-SETUP message is the remote gate information (from the received Dcs-Gate header where applicable), and any required electronic surveillance information

7.6.3.1.3 Anonymizer support

CMS_O MUST check the Dcs-Anonymity header, if present. If Full or IPAddr, CMS_O MUST provide IP address privacy through the use of an anonymizer service. CMS_O MUST update the Contact header with the anonymized address, and modify the "c=" line of the SDP with the anonymized data path address. Signaling between CMS_O and the anonymizer is described in Section 8.4. See Appendix Z for an example.

7.6.3.1.4 CMS/Proxy$_O$ actions handling 183-Session-Progress

$CMS/Proxy_O$ MUST collect the information needed to support mid-call changes to the call, and form a Dcs-State header. This information MUST include the gate location and Gate-ID at $CMTS_O$, needed to support mid-call codec changes, and SHOULD include the Request-URI after translation was completed, needed to reach the same destination endpoint for further requests. $CMS/Proxy_O$ MUST then take the set of all Dcs-State headers and form a single Dcs-State header containing them all. Using its private key, $CMS/Proxy_O$ MUST sign and encrypt the resulting Dcs-State header, and MUST replace all Dcs-State headers with this single Dcs-State header in the response formed to MTA_O.

CMS/Proxy$_O$ MUST add the Dcs-Media-Authorization header with the information for the gate at CMTS$_O$**.**

If the callee has requested privacy with Dcs-Anonymity: Full or URL, CMS/Proxy$_O$ MUST replace the URL in the Dcs-Remote-Party-ID header with a private URL and add "rpi-id=private". The private URL MUST be formed by encrypting the original URL and Dcs-Anonymity value (and possibly additional information) with CMS/Proxy$_O$'s privately held key, placing the resulting string as the username, inserting CMS/Proxy$_O$ as hostname, and adding a url-parameter of "private". If the callee has requested privacy with Dcs-Anonymity: Full or Name, CMS/Proxy$_O$ MUST delete the display-name in the Dcs-Remote-Party-ID header. If the caller has not subscribed to calling name delivery, then CMS/Proxy$_O$ MUST delete the display-name in the Dcs-Remote-Party-ID header. If the caller has not subscribed to calling number delivery, then CMS/Proxy$_O$ MUST replace the URL in the Dcs-Remote-Party-ID header with a private URL (as described above) and add "rpi-id=na." CMS/Proxy$_O$ MUST delete the Dcs-Anonymity header from the INVITE message.

The resulting 183-Session-Progress response is forwarded to MTA$_O$ as specified in the table below.

183-Session-Progress: (CMS/Proxy$_O$ -> MTA$_O$) Header:	***Requirement at CMS/Proxy$_O$***
SIP/2.0 183 Session Progress	*The Response line MUST be present.*
Via: SIP/2.0/UDP Host(MTA-o)	*MUST be copy of Via header in request from MTA$_O$*
Dcs-Remote-Party-ID:	*MUST be present; modified by CMS$_O$ based on privacy requested*
Dcs-State: Host(cms-o);{DS-o}$_K$	*MUST be present.* *MUST be added by CMS$_O$.* *MUST be encrypted with CMS$_O$'s private key.*
Dcs-Media-Authorization: GID-o	*MUST be added by CMS$_O$.*
From:	*MUST be present.* *MUST be copies of same headers in response from CMS$_T$*
To:	
Call-ID:	
CSeq:	
Contact: sip:Host(MTA-t | Ann-t)	*MUST be present.* *MUST be a copy of the Contact: header received in the response from CMS$_T$ as modified for IP privacy.*
Session: qos	*MUST be present. MUST containe 'qos' and MAY also contain 'security'. MUST NOT contain 'media'*
Rseq: x_t	*MUST be present.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (...)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= s= o= m= c= a=	*MUST be present.* *SDP description of media streams acceptable to the destination endpoint, as described in Section 5.* *MUST contain a line 'a=X-pc-qos:mandatory' with attribute 'confirm'*

For CMS/Proxy$_O$ there is no retransmission of this message, as EP$_T$ performs the required retransmissions and receives the PRACK message. CMS/Proxy$_O$ MUST store the 183-Session-Progress response for a period of time (32 seconds[11]) to handle retransmissions of the 183-Session-Progress from CMS$_T$. At the end of this period, CMS/Proxy$_O$ MAY delete all its state information related to this call and process all remaining provisional and final responses as a stateless proxy.

7.6.3.1.5 CMS/Agent$_O$ actions handling 183-Session-Progress

If the 183-Session-Progress provisional response was the first response to the sent INVITE, CMS/Agent$_O$ MUST cancel the retransmission timer (T1), and MUST set the transaction timer (T3) for this exchange to T-setup. The default value of (T-setup) is given in Appendix A. On expiration of T3, CMS/Agent$_O$ MUST

clear the call attempt, and send a CANCEL message to CMS_T with the same values of Request-URI, From, To, and Call-ID, and any Dcs-State headers received for this call attempt, as shown in 7.6.9.

$CMS/Agent_O$ stores the Dcs-State headers, the Contact header and the SDP description for the duration of the call. The Dcs-State header values received in the 183-Session-Progress MUST be included in all requests and responses sent to CMS_T for this call leg.

$CMS/Agent_O$ MUST send a PRACK to acknowledge receipt of the 183-Session-Progress. The PRACK message MUST be sent directly to the address specified in the Contact header of the received 183-Session-Progress.

An SDP MUST be included in the PRACK message. The SDP in the PRACK MUST include a media (m=) line with a single CODEC to be used for this connection.

PRACK: ($CMS/Agent_O$ -> EP_T) Header:	***Requirement at CMS_O for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the most recent Contact header received*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in the provisional response.*
Call-ID:	
Cseq: n_o+1 PRACK	*Sequence number MUST be one higher than initial INVITE, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be presen.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present* *Contains the SDP description as modified after processing the SDP returned by the terminating endpoint, and MUST contain a single CODEC choice.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK. The 200-OK response MUST be as follows.

200-OK Header: (EP_T -> $CMS/Agent_O$) Header:	***Requirement on CMS_O for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.* *Method in Cseq MUST be PRACK.*
To:	
Call-ID:	
Cseq:	

Following receipt of the 183-Session-Progress response, $CMS/Agent_O$ tells the endpoint device to attempt to reserve access network resources based on the SDP parameters sent in the PRACK message. After

successful completion of the resource reservation, CMS/Agent$_O$ MUST send a PRECONDITION-MET message to EP$_T$. This informs the destination that resources are available and that it may proceed and alert the end user. The PRECONDITION-MET message MUST be as follows.

PRECONDITION-MET: (CMS/Agent$_O$ -> EP$_T$) Header:	***Requirement at CMS$_O$ for message generation***
PRECONDITION-MET SIP-URL SIP/2.0	*MUST be present. Method MUST be PRECONDITION-MET. The value of the SIP-URL MUST be the most recent Contact header received*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS$_O$.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE.*
Call-ID:	
Cseq: n_o+2 PRECONDITION-MET	*Sequence number MUST be one higher than the last sequence number sent by CMS$_O$, method MUST indicate PRECONDITION-MET*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be presen.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present* *Contains the SDP description as modified after processing the SDP returned by the terminating endpoint, and MUST contain a single CODEC choice.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T- direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The originating endpoint SHOULD be prepared to receive bearer channel packets once CMS/Agent$_O$ has transmitted the PRECONDITION-MET.

The 200-OK response to the PRECONDITION-MET MUST be as follows.

200-OK: (EP$_T$ -> CMS/Agent$_O$) Header:	***Requirement on CMS$_O$ for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRECONDITION-MET message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRECONDITION-MET message.*
To:	
Call-ID:	
Cseq:	

If the resource reservation fails, CMS/Agent$_O$ SHOULD send a CANCEL to CMS$_T$.

CANCEL: (CMS$_O$ -> CMS$_T$) Header:	***Requirement at CMS$_O$ for message generation***
CANCEL DCS-URL SIP/2.0	*MUST be present. Method MUST be CANCEL. The value of the DCS-URL MUST be the Request-URI used in the initial INVITE*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS$_O$.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE.*

Call-ID:	
Cseq: n_o+2 CANCEL	*Sequence number MUST be one higher than the last sequence number sent by CMS_O, method MUST indicate CANCEL*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The 200-OK response to the CANCEL MUST be as follows.

200-OK: (CMS_T -> CMS_O) Header:	***Requirement on CMS_O for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the CANCEL message*
From:	*From, To, CallID, and Cseq headers MUST match those of the CANCEL message.*
To:	
Call-ID:	
Cseq:	

7.6.3.2 302-Redirect status response handling at CMS_O

CMS_O MUST check the headers of a 302-Redirect response as follows.

302-Redirect (CMS_T -> CMS_O) Header:	***Requirement on CMS_O for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST bepresentt. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be identical to the INVITE message*
Dcs-State:	*MUST be present if CMS_O sent Dcs-State in INVITE message*
Dcs-Billing-ID:	*MUST be identical to the INVITE message*
Dcs-Billing-Info: Host(rks-o):Port(rks-o)<AC-o/E.164-o/E.164-t>	*MUST be identical to the INVITE message*
Dcs-Billing-Info: Host(rks-t):Port(rks-t)<AC-t/E.164-t/E.164-f>	*A second Dcs-Billing-Info header MUST be present*
Dcs-Laes:	*MAY be present*
Dcs-Remote-Party-ID:	*MUST be present.*
Dcs-Anonymity:	*MAY be present*
From:	*From, To, CallID, and Cseq headers MUST be identical to the INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be present, and MUST be a valid URI..*
Expires:	*MAY be present*

CMS_O MUST match the 302-Redirect response to the INVITE it had sent out. CMS_O MUST send an ACK back to CMS_T, using the same Request-URI from the INVITE request that was previously forwarded to CMS_T.

ACK: (CMS_O -> CMS_T) Header:	***Requirement at CMS_O for message generation***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. DCS-URL MUST be a copy of DCS-URL in request line from the original request forwarded to CMS_T.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present.* *MUST be copies of same headers in redirect response from CMS_T.*
To:	
Call-ID:	

Cseq: n_0 ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE, method MUST indicate ACK*

Following sending of the ACK message to CMS_T, CMS_O MUST reissue an INVITE request to the party indicated by the Contact header in the Redirect response. CMS_O MUST attempt to resolve the DCS-URL from the Contact header into a destination IP address.

If CMS_O performs the local-number-portability lookup, it MUST generate a Request-URI containing a DCS-URL with the user-param "user=np-queried." CMS_O MUST include the lrn-tag indicating the returned value if the local-number-portability lookup returned a value.

If the destination endpoint is not served by CMS_O, CMS_O generates an INVITE message and sends it to CMS_F, the CMS that manages the forwarded-to destination.

If a Dcs-LAES header is present in the 3xx response, CMS_O MUST include that header unchanged in the reissued INVITE. CMS_O MUST also include a Dcs-Redirect header containing the original dialed number, the new destination number, and the number of redirections that have occurred.

The rest of the INVITE message MUST appear identical to that which was sent to CMS_T, with the exception of an additional Dcs-Billing-Info header. CMS_O MUST copy the Dcs-Billing-Info header from the Redirect response so that the forwarding party can be billed for the newly created leg of the call.

The format of the resulting INVITE message as sent by CMS_O to CMS_F and the associated requirements on the header fields are as follows:

INVITE: (CMS_O -> CMS_F) Header:	***Additional requirements for message generation***
INVITE DCS-URL SIP/2.0	*As described above*
Via: SIP/2.0/UDP Host(cms-o);branch=m	*MUST contain the IPAddress or FQDN of CMS_O.* *MUST include a branch parameter, with a different value than the previous INVITE.*
Via: SIP/2.0/UDP Host(mta-o)	*CMS/Proxy only: As described in 7.6.1.*
Dcs-Remote-Party-ID:	*As described in 7.6.1.*
Dcs-Anonymity:	*As described in 7.6.1.*
Dcs-Gate:	*As described in 7.6.1.*
Dcs-Billing-ID:	*As described in 7.6.1.*
Dcs-Billing-Info:	*MUST include all Dcs-Billing-Info provided by CMS_T in Redirect response*
Dcs-State:	*SHOULD be present, as described in 7.6.1.*
Dcs-Laes:	*As described above*
Dcs-Redirect:	*As described above*
From:	*As described in 7.6.1.*
To:	*As described in 7.6.1.*
Call-ID:	*As described in 7.6.1.*
CSeq: n_0 INVITE	*As described in 7.6.1.*
Contact:	*As described in 7.6.1.*
---all other headers provided by MTA to CMS/Proxy---	*CMS/Proxy only: As described in 7.6.1.*
Content-Type: application/sdp	*As described in 7.6.1.*
Content-length: (…)	*As described in 7.6.1.*
	As described in 7.6.1.
v= o= s= c= b= t= a= m=	*As described in 7.6.1.* *c= line MAY be modified in support of IP address privacy* *CMS_O MUST add a "a=X-pc-secret" line to the SDP, giving a security key to be used by the media packets for this session (e.g. clear:RC4/ItWasTheBestOfTimesItWasTheWorstOfTimes), unless one was supplied by originating endpoint.*

On recipt of this INVITE message, CMS_F uses the combination of From:, To:, Call-ID:, and Request-URI to recognize this as a new call and not a retransmission from a previous call.

The behavior and processing of the INVITE at CMS_F is identical to that described in section 7.6.2.

7.6.4 CMS/Agent$_T$ Receiving Acknowledgement of 183-Session-Progress

This subsection applies only to CMS/Agent$_T$, as a CMS/Proxy is not involved in the protocol exchanges described here.

After sending the 183-Session-Progress response to the INVITE, CMS/Agent$_T$ MUST wait for the PRACK message acknowledging the Session-Progress. The PRACK message headers MUST be checked as follows.

PRACK: (EP$_O$ -> CMS/Agent$_T$) Header:	***Requirement at CMS/Agent$_T$ for message checking***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the most recent Contact header received*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE request.*
Call-ID:	
Cseq: n_o+1 PRACK	*Sequence number 'n_o+1' MUST be one higher than sequence number in INVITE, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present.* *MUST be as defined in 4.6.4..*
Content-length: (...)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after processing the SDP returned by the terminating MTA, and MUST contain a single CODEC choice.*

On receipt of this PRACK, CMS/Agent$_T$ MUST respond with a 200-OK. The 200-OK response MUST be as follows.

200-OK: (CMS/Agent$_T$ -> EP$_O$) Header:	***Requirement on CMS$_T$ for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	
Cseq:	

Following receipt of the PRACK message, CMS/Agent$_T$ instructs the endpoint to reserve network resources. The resource reservation request is based on the SDP parameters received in the PRACK message.

After the originating endpoint successfully completes the resource reservation, it sends a PRECONDITION-MET message to CMS/Agent$_T$. This informs CMS/Agent$_T$ that resources are available and that it may proceed and alert the end user. CMS/Agent$_T$ MUST check and verify the PRECONDITION-MET message as follows.

PRECONDITION-MET: (EP$_O$ -> CMS/Agent$_T$) Header:	***Requirement at CMS$_T$ for message checking***
PRECONDITION-MET SIP-URL SIP/2.0	*MUST be present. Method MUST be PRECONDITION-MET. The value of the SIP-URL MUST be the most recent Contact header received*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE request.*
Call-ID:	
Cseq: n_o+2 PRECONDITION-MET	*MUST indicate PRECONDITION-MET*
Content-Type: application/sdp	*MUST be present.* *MUST be as defined in 4.6.4.*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present.* *Contains the SDP description as modified after processing the SDP returned by the terminating MTA, and MUST contain a single CODEC choice.*

CMS/Agent$_T$ MUST respond to the PRECONDITION-MET request with a 200-OK. The 200-OK response to the PRECONDITION-MET MUST be as follows.

200-OK: (CMS/Agent$_T$ -> EP$_O$) Header:	***Requirement on CMS$_T$ for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRECONDITION-MET message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRECONDITION-MET message.*
To:	
Call-ID:	
Cseq:	

On receipt of the PRECONDITION-MET message, and successfully reserving the network resources needed for its media flows, CMS/Agent$_T$ MUST cancel timer T3, and continue with the alerting procedures of section 7.6.5.

If the resource reservation fails, CMS/Agent$_T$ MUST send a 580-Precondition-Failure response to CMS$_O$.

580-Precondition-Failure: (CMS$_T$ -> CMS$_O$) Header:	***Requirement on CMS$_T$ for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*

Via:	*MUST be copied from the INVITE message*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From, To, CallID, and Cseq headers MUST match those of the INVITE message.*
To:	
Call-ID:	
Cseq:	

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

7.6.5 CMS_T sends 180-Ringing/183-Media

Once EP_T receives the PRECONDITION-MET message, and resource reservation is successful, CMS_T MUST send a provisional or final response to the originating endpoint, through the proxy path taken by the initial INVITE request.

$CMS/Agent_T$ determines, by mechanisms beyond the scope of this specification, whether alerting is necessary, and if so whether media will be generated to perform the "ringback" function. If alerting of the destination user is necessary, and the destination desires to provide the "ringback tone" to the call originator, $CMS/Agent_T$ sends a 183-Session-Progress(Media) response. If alerting only of the destination user is necessary, $CMS/Agent_T$ sends a 180-Ringing response. Otherwise, $CMS/Agent_T$ sends a final response as described in section 7.6.7.

$CMS/Proxy_T$ determines whether alerting is necessary or not by the response sent by MTA_T. Receipt of a 180-Ringing response indicates alerting is necessary. Receipt and handling of a final response is described in section 7.6.7.

The 180-Ringing/183-Session-Progress(Media) message MUST be as follows:

18x Ringing: (CMS_T -> CMS_O) Header:	***Requirement***
SIP/2.0 180 Ringing Or SIP/2.0 183 Session-Progress	*Status line with status code 18x MUST be present.*
Via:	*MUST be present and copied from INVITE message.*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the received INVITE. This triple identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present and copied from initial 183-Session-Progress.*
Cseq:	*MUST be present. MUST be the same as that in the received INVITE. Method MUST be INVITE. Identifies the message which caused this response.*
Session: Media	*MUST be present in 183-Session-Progress message, and MUST contain "media"* *MUST NOT be present in 180-Ringing message*
RSeq: x_t+1	*MUST be present. MUST contain value one greater than previous provisional response sequence number*

For $CMS/Proxy_T$ there is no retransmission of this message, as MTA_T performs the required retransmissions, receives the PRACK message, and acknowledges the PRACK with a 200-OK. No further action is required of a $CMS/Proxy_T$. $CMS/Proxy_T$ MAY store the message for a period of time (32 seconds[11]) to optimize retransmissions from MTA_T.

For CMS/Agent$_T$, the retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of PRACK.

On sending the status 180-Ringing/183-Session-Progress(Media) message, CMS/Agent$_T$ MUST start the transaction timer (T3) with value T-ringing. The default value of (T-ringing) is given in Appendix A. Timer T3 is canceled by the user indicating call acceptance, or receipt of a BYE or CANCEL request. On expiration of timer T3, CMS/Agent$_T$ MUST either perform call-forwarding-no-answer, or send a 480-Temporarily-Unavailable response to CMS$_O$.

After sending the 180-Ringing response to the INVITE, CMS/Agent$_T$ MUST wait for the PRACK message acknowledging the response. The PRACK message headers MUST be checked as follows.

PRACK: (EP$_O$ -> CMS/Agent$_T$) Header:	***Requirement at CMS/Agent$_T$ for message checking***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the most recent Contact header received*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE request.*
Call-ID:	
Cseq: n_o+2 PRACK	*Sequence number MUST be one higher than sequence number in previous request, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 180-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*

On receipt of this PRACK, CMS/Agent$_T$ MUST respond with a 200-OK. The 200-OK response MUST be as follows.

200-OK: (CMS/Agent$_T$ -> EP$_O$) Header:	***Requirement on CMS/Agent$_T$ for message generation***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	
Cseq:	

7.6.6 CMS$_O$ receives 180-Ringing/183-Media

After the originating endpoint has completed the resource reservation, and it (or its CMS/Agent) has sent the PRECONDITION-MET message to the destination endpoint, CMS$_O$ will receive either (1) a provisional response of 180-Ringing or 183-Session-Progress(Media), (2) a final response of 200-OK or (3) an error. This section covers the procedures for the provisional responses, 180 and 183, and section 7.6.8 covers the procedures for the final responses.

CMS$_O$ MUST verify the headers of the provisional response according to the following table.

18x Provisional Response (CMS$_T$ -> CMS$_O$) Header:	***Requirement for checking at CMS$_O$***

SIP/2.0 180 Ringing Or SIP/2.0 183 Session Progress	*Status line with status code 180 or 183 MUST be present.*
Via:	*MUST be present and match that in INVITE message.*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.* *MAY include Dcs-State header from CMS_T*
From:	*From:, To: and Call-ID MUST be present and MUST match those in the initial INVITE. This triple identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present. MUST be same as in 183-Session-Progress*
Cseq: n_o INVITE	*MUST be present. MUST be the same as that in the initial INVITE. Method MUST be INVITE. Identifies the message which caused this response.*
Session: Media	*MUST be present for 183-Session-Progress, MUST contain 'Media' MUST NOT be present for 180-Ringing*
Rseq: x_t+1	*MUST be present. MUST be value one greater than most recent provisional response Rseq value*

Upon receipt of the 18x message, $CMS/Agent_O$ MUST cancel the T3 session timer. A CMS/Agent MUST restart the session timer T3 with the value T-ringback. The default value of (T-ringback) is given in Appendix A. The T3 timer MUST be cancelled on receipt of 200-OK or other final response.

The 180-Ringing response indicates to the originating MTA that it should supply a local ringback. The 183-Session-Progress response indicates that the ringback is supplied via audio packets from the data network. $CMS/Agent_O$, by methods outside the scope of this specification, informs the originating endpoint of the desired actions.

$CMS/Proxy_O$ MUST send the 18x provisional response to MTA_O, as described in section 6.4.6.

$CMS/Agent_O$ MUST acknowledge the 18x provisional response with a PRACK message, as described in the following table.

PRACK: ($CMS/Agent_O$ -> EP_T) **Header:**	***Requirement at $CMS/Agent_O$ for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the most recent Contact header received s*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present.* *MUST be copies of same headers in initial INVITE.*
To:	
Call-ID:	
Cseq: n_o+3 PRACK	*Sequence number MUST be one higher than sequence number in the latest request, method MUST indicate PRACK*
Rack: x_t n_o INVITE	*Value 'x_t' MUST be a copy of the value in the Rseq header of the 183-Session-Progress. Value 'n_o' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T- direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK. The 200-OK response MUST be as follows.

200-OK: (EP_T -> $CMS/Agent_O$) **Header:**	***Requirement on $CMS/Agent_O$ for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be identical to the PRACK message*
From:	*From, To, CallID, and Cseq headers MUST match those of the PRACK message.*
To:	
Call-ID:	

Cseq:	

7.6.7 CMS_T Sending final Response

After the destination endpoint has successfully reserved resources, EP_T has received the PRECONDITION-MET message from EP_O (indicating it had also successfully reserved resources), and the destination endpoint has completed whatever alerting procedures were required, CMS_T sends a final response. For a typical telephony service, this is indicated by the user 'going offhook' and 'answering the phone', and means the endpoint is ready to begin media transfers. The case of a successful completion of a call is covered in section 7.6.7.1, and the various error cases are covered in section 7.6.7.2 and 7.6.7.3.

7.6.7.1 CMS_T sending 200-OK Final Response

Once CMS_T determines that the destination endpoint is willing to accept the incoming call (e.g. off-hook or hook-flash), it MUST send a 200-OK status message directly to the originating MTA. A CMS/Agent makes this determination by methods beyond the scope of this specification, while a CMS/Proxy receives a 200-OK response from MTA_T as described in section 6.4.7.1. The message sent by CMS_T to CMS_O MUST be as follows:

200-OK: (CMS_T -> CMS_O) Header:	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.* *MAY contain a Dcs-State header from CMS_T*
From:	*From:, To: and Call-ID MUST be present and MUST be a copy of the initial INVITE message.*
To:	
Call-ID:	
CSeq:	*MUST be present.* *MUST be the same as in the initial INVITE message.*

On sending the 200-OK, CMS/$Agent_T$ MUST stop timer T3, tell the endpoint device to commit to resources that have been reserved for this call, and tell the endpoint device that it MAY begin sending bearer channel packets.

The terminating device SHOULD be prepared to receive bearer channel packets once it has sent a final response.

For CMS/$Proxy_T$ there is no retransmission of this message, as MTA_T performs the required retransmissions, and receives the ACK message. No further action is required of CMS/$Proxy_T$.

For CMS/$Agent_T$, the retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

The ACK message header fields MUST be verified as follows:

ACK: (EP_O -> CMS/$Agent_T$) Header:	***Requirement at CMS/$Agent_T$ for checking message***

ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value in the most recent Contact header received.*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in initial INVITE request.*
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

7.6.7.2 CMS_T sending 3xx-Redirect Final Response

If the terminating endpoint wishes to forward the call (e.g. if call-forwarding-no-answer is enabled), a final 3xx-Redirect status response MUST be sent by CMS_T, with the forwarded-to destination URI in the contact header. $CMS/Agent_T$ determines this by means beyond the scope of this specification. $CMS/Proxy_T$ determines this by a 302-Redirect final response from MTA_T, as described in 6.4.7.2.

If the destination endpoint is not a subscriber to the Call Forwarding service, CMS_T MUST send a 480-Temporarily-Unavailable error response to CMS_O, using the procedures described in section 7.6.7.3.

CMS_T MUST check for an outstanding lawfully authorized surveillance order for the terminating subscriber. If found, CMS_T MUST include a Dcs-LAES header in the 3xx-Redirect response. The Dcs-LAES header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be delivered, and MUST include a random string for use as a security key between the Delivery Functions.

CMS_T MUST add a Dcs-Billing-Info header to the response to allow the "second leg" of the forwarded call to be charged to the terminating party, and MUST copy the Dcs-Billing-Info and Dcs-Billing-ID provided by CMS_O in the request..

CMS_T MUST forward the following 3xx REDIRECT response to CMS_O.

302-Redirect: (CMS_T -> CMS_O) Header:	***Requirement on CMS_T for message generation***
SIP/2.0 302 Moved Temporarily	*Status line header MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-Billing-ID:	*MUST be copied from the INVITE message*
Dcs-Billing-Info: Host(rks-o):Port(rks-o)<AC-o/E.164-o/E.164-t>	*MUST be copied from the INVITE message*
Dcs-Billing-Info: Host(rks-t):Port(rks-t)<AC-t/E.164-t/E.164-f>	*New Billing-Info header as described above*
Dcs-Laes:	*MAY be present, as described above*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.* *MAY contain a Dcs-State header from CMS_T*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be present, and carries the new destination information.* *MUST be a valid URI.* *If the new destination is a telephone number, then the format of the URI MUST be a tel: URL where the URL contains a full E.164 number.*
Expires:	*MAY be present*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

ACK: (CMS_O -> CMS_T) Header:	***Requirement at CMS_T***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack. Request-URI MUST be copy of the initial INVITE*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present. MUST be copies of same headers in initial INVITE.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST same as initial INVITE. Method MUST indicate ACK*

7.6.7.3 Other Status Response to INVITE request

A final error response, 4xx, 5xx, or 6xx response, MUST be sent as per [11]. This includes, but is not limited to, 480-Temporarily-Unavailable. The error response MUST be generated as follows.

Error: (CMS_T -> CMS_O) Header:	***Requirement on CMS_T for message generation***
SIP/2.0 xxx	*Status line header MUST be present. MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T-proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of ACK.

ACK: (CMS_O -> CMS_T) Header:	***Requirement at CMS_T for message checking***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack. Request-URI MUST be copy of initial INVITE.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present. MUST be copies of same headers in initial INVITE message.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE message. Method MUST indicate ACK*

7.6.8 CMS_O Receives final response from CMS_T

7.6.8.1 CMS_O Receiving 200-OK

Once the terminating endpoint is willing to accept the incoming call (e.g. off-hook or hook-flash), it sends a 200-OK status message to the originating endpoint, via the proxy-proxy signaling path. The message sent by CMS_T to CMS_O MUST be as follows.

200-OK (CMS_T -> CMS_O) Header:	***Requirement for CMS_O for message checking***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
Dcs-State:	*MUST be present if Dcs-State header present in INVITE message sent to CMS_T*
From:	*From:, To: and Call-ID MUST be present and MUST be identical to the initial INVITE message.*
To:	
Call-ID:	
Cseq:	*MUST be present. MUST be the same as in the INVITE.*

On receiving the final response, $CMS/Proxy_O$ MUST forward a final response to MTA_O, as described in 6.4.8.1.

On receiving the final response, $CMS/Agent_O$ MUST stop timer T3, tell the endpoint device to commit to resources that have been reserved for this call, and tell the endpoint device that it SHOULD begin sending bearer channel packets.

$CMS/Agent_O$ MUST acknowledge the 200-OK response with an ACK message. The header fields MUST be generated as follows:

ACK: ($CMS/Agent_O$ -> EP_T) Header:	***Requirement at CMS_O for message generation***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the most recent Contact header.*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in initial INVITE request.*
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

7.6.8.2 CMS_O receiving 302-Redirect

If the terminating device wished to forward the call (e.g. if call-forwarding-no-answer was enabled at the destination), a 302-Redirect status response is sent back through the CMS/Proxies with the forwarded-to destination URI in the contact header. The message sent by CMS_T to CMS_O MUST be as follows.

302-Redirect: (CMS_T -> CMS_O) Header:	***Requirement on CMS_O for message checking***
SIP/2.0 302 Moved Temporarily	*Status line header MUST be present. MUST include the SIP version number and the three digit status code.*
Via:	*MUST be identical to the INVITE message*
Dcs-Laes:	*MAY be present.*

Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From, To, CallID, and Cseq headers MUST be identical to the INVITE message.*
To:	
Call-ID:	
Cseq:	
Contact: URI	*MUST be present. Carries the new destination information. MUST be a valid URI. MUST NOT have a private-param url-parameter.*
Expires:	*MAY be present*

CMS_O MUST send an ACK message to CMS_T. The required fields of the message are as shown below.

ACK: (CMS_O -> CMS_T) Header:	***Requirement at CMS_O for message generation***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack. Request-URI MUST be copy of initial INVITE.*
Via:	*MUST be present. MUST contain the Ipaddress or FQDN of CMS_O*
From:	*MUST be present. MUST be copies of same headers in initial INVITE request.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE request, method MUST indicate ACK*

In response to a 302-Redirect final response, CMS/$Proxy_O$ MUST send the 302-Redirect response to MTA_O, as described in section 6.4.8.2. The Contact header in this message MUST be a private URL, formed from the following information: 1) the destination E.164 address, 2) the sequence of Dcs-Billing-Info values, which indicate the complex charging arrangement for the new call, 3) an expiration time very shortly in the future, to limit the ability of MTA_T to re-use this private-param for multiple calls, and 4) the electronic surveillance information, if present. CMS/$Proxy_O$ MAY ignore the value given by the destination in the Expires: header in determining the expiration time of the private URL.

Following sending of the ACK message to CMS_T, CMS/$Agent_O$ MUST reissue an INVITE request to the party indicated by the Contact header in the Redirect response. CMS/$Agent_O$ MUST attempt to resolve the DCS-URL from the Contact header into a destination IP address.

If CMS/$Agent_O$ performs the local-number-portability lookup, it MUST generate a Request-URI containing a DCS-URL with the user-param "user=np-queried." CMS/$Agent_O$ MUST include the lrn-tag indicating the returned value if the local-number-portability lookup returned a value.

If the destination endpoint is not served by CMS/$Agent_O$, CMS/$Agent_O$ generates an INVITE message and sends it to CMS_F, the CMS that manages the forwarded-to destination.

If a Dcs-LAES header is present in the 3xx response, CMS/$Agent_O$ MUST include that header unchanged in the reissued INVITE. CMS/$Agent_O$ MUST also include a Dcs-Redirect header containing the original dialed number, the new destination number, and the number of redirections that have occurred.

The rest of the INVITE message MUST appear identical to that which was sent to CMS_T, with the exception of an additional Dcs-Billing-Info header. CMS/$Agent_O$ MUST copy the Dcs-Billing-Info header from the Redirect response so that the forwarding party can be billed for the newly created leg of the call.

The format of the resulting INVITE message as sent by CMS/$Agent_O$ to CMS_F and the associated requirements on the header fields are as follows:

INVITE: (CMS_O -> CMS_F) Header:	***Additional requirements for message generation***
INVITE DCS-URL SIP/2.0	*As described above*

Via: SIP/2.0/UDP Host(cms-o);branch=m	*MUST contain the IPAddress or FQDN of CMS_O.* *MUST include a branch parameter, with a different value than the previous INVITE.*
Dcs-Remote-Party-ID:	*As described in 7.6.1.*
Dcs-Anonymity:	*As described in 7.6.1.*
Dcs-Gate:	*As described in 7.6.1.*
Dcs-Billing-ID:	*As described in 7.6.1.*
Dcs-Billing-Info:	*MUST include all Dcs-Billing-Info provided by CMS_T in Redirect response*
Dcs-State:	*SHOULD be present, as described in 7.6.1.*
Dcs-Laes:	*As described above*
Dcs-Redirect:	*As described above*
From:	*As described in 7.6.1.*
To:	*As described in 7.6.1.*
Call-ID:	*As described in 7.6.1.*
CSeq: n_0 INVITE	*As described in 7.6.1.*
Contact:	*As described in 7.6.1.*
Content-Type: application/sdp	*As described in 7.6.1.*
Content-length: (…)	*As described in 7.6.1.*
	As described in 7.6.1.
v= o= s= c= b= t= a= m=	*As described in 7.6.1.* *c= line MAY be modified in support of IP address privacy* *CMS_O MUST add a "a=X-pc-secret" line to the SDP, giving a security key to be used by the media packets for this session (e.g. clear:RC4/ItWasTheBestOfTimesItWasTheWorstOfTimes) if not provided by endpoint*

On receipt of this INVITE message, CMS_F uses the combination of From:, To:, Call-ID:, and Request-URI headers to recognize this as a new call and not a retransmission from a previous call.

The behavior and processing of the INVITE at CMS_F is identical to that described in section 7.6.2.

7.6.8.3 CMS_O receiving other error response

A final error response, 4xx, 5xx, or 6xx response, MAY be sent as per [11]. This includes, but is not limited to, 480-Temporarily-Unavailable. The error response MUST be verified as follows.

Error: (CMS_T -> CMS_O) **Header:**	***Requirement on CMS_O for message checking***
SIP/2.0 xxx	*Status line header MUST be present.* *MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the INVITE message*
Dcs-State:	*If Dcs-State header was present in INVITE message from CMS_O, then MUST be present and MUST be copy of that value.*
From:	*From, To, CallID, and Cseq headers MUST be copied from INVITE message.*
To:	
Call-ID:	
Cseq:	

CMS_O MUST send an ACK message to acknowledge the error response.

ACK : (CMS_O -> CMS_T) **Header:**	***Requirement at CMS_O for message generation***
ACK DCS-URL SIP/2.0	*The Response line MUST be present. Method MUST be ack.* *Request-URI MUST be copy of initial INVITE.*

Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present.* *MUST be copies of same headers in initial INVITE message.*
To:	
Call-ID:	
Cseq: n_o ACK	*Sequence number MUST be copy of CSEQ value in initial INVITE message. Method MUST indicate ACK*

7.6.9 Session Timer expiration at CMS_O

On expiration of timer T3, CMS_O SHOULD send a CANCEL request to CMS_T, and MUST release all resources reserved for this connection. The CANCEL request MUST be as described below.

CANCEL: (CMS_O -> CMS_T) Header:	***Requirement at CMS_O for message generation***
CANCEL DCS-URL SIP/2.0	*MUST be present. Method MUST be CANCEL. The value of the DCS-URL MUST be the original Request-URI used in the INVITE message.*
Via:	*MUST be present. MUST be the IP address or FQDN of CMS_O.*
From:	*MUST be present.* *MUST be copies of same headers in Request from CMS_O.*
To:	
Call-ID:	
Cseq:	*Sequence number MUST be one higher than the last sequence number sent by CMS_O, method MUST indicate CANCEL*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

The 200-OK response to the CANCEL MUST be as follows.

200-OK: (CMS_T -> CMS_O) Header:	***Requirement on CMS_O for message checking***
SIP/2.0 200 OK	*Status line header MUST be present. It MUST include the SIP version number and the three digit status code.*
Via:	*MUST be copied from the CANCEL message*
From:	*From, To, CallID, and Cseq headers MUST match those of the CANCEL message.*
To:	
Call-ID:	
Cseq:	

7.7 Initiating a 9-1-1 call

A call for emergency services, e.g. 9-1-1, MUST follow the procedures given for a basic call, as given in section 7.6, with the following exceptions.

If the originating endpoint is not authorized for outgoing service, CMS_O MAY permit the call to the emergency services number.

If the Dcs-Remote-Party-ID header is absent or invalid, and CMS_O is unable to establish the originator of the call, CMS_O MAY permit the call to the emergency services number.

CMS_O, receiving a 183-Session-Progress response for a 9-1-1 call, MUST indicate enhanced priority for access network admission control in the GATE-SETUP command to the originating CMTS, extending the procedure described in 7.6.3.1.2.

$CMS/Agent_O$ MUST disable the call waiting feature, so that any incoming call to the endpoint is given a BUSY error instead of call-waiting treatment.

If the endpoint desires to terminate the session, $CMS/Agent_O$ MUST NOT sent a BYE request to the Emergency Services Center; rather $CMS/Agent_O$ MUST wait for a BYE request to initiate at the Emergency Services Center.

7.8 CMS handling of Mid-Call Changes

Mid-call changes include call-hold, call-resume, call transfer, ad-hoc conferencing (e.g. three-way calling), and dynamic codec changes. For CMS/Proxies, these are intiated by the MTA, while they are generated directly by CMS/Agents. Since the handling is different in each case, the first subsection specifies the handling for CMS/Proxies, and following subsections detail the requirements for CMS/Agents.

The initiator of a mid-call change in this section is referred to as MTA_I or $CMS/Agent_I$, and the recipient of a mid-call change is referred to as MTA_R or $CMS/Agent_R$.

Dcs-Also and Dcs-Replaces provide tools by which many call control services are built. For purposes of this specification, only three are completely specified at the initiator: blind transfer, consultative transfer, and ad-hoc conferencing. The procedures necessary to support these are completely specified at the recipient. Based on knowledge of the recipient behavior, the originator MAY perform many other complex call control operations, beyond those specified here.

Not every function described in this section is applicable to every CMS/Agent. For example, a Media-Gateway-Controller will not likely initiate consumer endpoint services, such as three-way calling. Testing of the requirements given in this section must therefore be dependent on the architectural function being implemented by the specific CMS.

7.8.1 CMS/Proxy handling of Mid-Call Changes

MTA_I MAY send an INVITE to its CMS/Proxy during an active call to request a change in its QoS authorization, or to request a special call-handling feature. Examples of such features are Call-Transfer, Three-Way-Calling, and CODEC change.

All INVITE messages from MTA_I that modify existing calls MUST have a Dcs-State header, from which $CMS/Proxy_I$ retrieves the local state information needed to process the mid-call change.

An INVITE request that includes “Dcs-Also:” and/or “Dcs-Replaces:” headers is typically used to add and/or remove parties in a call in progress. The “Dcs-Also:” header indicates parties to be added and the “Dcs-Replaces:” header indicates parties to be removed. These two SIP header extensions are defined in Sections 3.3.7. Because these functions involve modifying the participants in a call, the requests are sent via CMS/Proxies.

In implementing Call-Transfer, MTA_I will send an INVITE (also, replace). In implementing the Three-Way-Calling feature, MTA_I will send an INVITE(Also) to establish the connections to a conference bridge. The conference bridge will send INVITE(replace) to the parties to redirect their connections to the bridge. Upon hangup of a participant in a three-way-call, the bridge will send INVITE(also,replace) to the remaining participant to revert to a two-party call.

In addition to the general requirements for headers, as given in Section 7.5, the following MUST be satisfied:

INVITE(mid-call-change): (MTA_I -> $CMS/Proxy_I$) Header:	***Requirement at CMS/Proxy***
INVITE SIP-URL SIP/2.0	*The Request line MUST be present.* *The Request-URI MUST be copied from the most recent Contact header* *Note: Routing of this request is based on the saved Dcs-State header values, and not on the Request-URI.*
Via: SIP/2.0/UDP Host(MTA-i)	*MUST be present.* *MUST represent same party as Dcs-Remote-Party-ID: header.*
Dcs-State:	*MUST be present.*
Dcs-Remote-Party-ID:	*SHOULD be present*
From:	*MUST be copy of either From or To header from original INVITE request.*
To:	*MUST be copy of either From or To header from original INVITE request*
Call-ID: ID	*The Call-ID MUST be the same as the Call-ID from the INVITE request*
CSeq: n+1 INVITE	*MUST be present.*
Dcs-Anonymity:	*MAY be present.*
Dcs-Also: DCS-URL	*MAY be present. If present, MUST conform to the rules for DCS-URLs stated in section 4.2*
Dcs-Replaces: SIP-URL	*MAY be present. If present, MUST conform to the rules for SIP-URLs stated in section 4.2*

In addition, an SDP MAY be present as a request for a change in resource authorization.

$CMS/Proxy_I$ MUST verify that the Via header represents MTA_I, via the IPSec security association.

If both Dcs-Also and Dcs-Replaces headers are present, $CMS/Proxy_I$ MUST verify that MTA_I has subscribed to Call Control services.

If any of these checks fail, $CMS/Proxy_I$ MUST respond with an appropriate 4xx, 5xx, or 6xx error code.

If the message passes the above checks, $CMS/Proxy_I$ MUST decrypt the Dcs-State information to determine the local gate location and identification, as well as the Request-URI for reaching CMS_R.

After performing the processing needed for the individual headers, as described in the following sections, the request is passed from $CMS/Proxy_I$ to CMS_R, processed at $CMS/Proxy_R$ as needed for the individual headers, and then passed to MTA_R, following the routing contained in saved CMS state, or in Dcs-State header values.

7.8.1.1 CMS/Proxy Handling of Dcs-Replaces

The Dcs-Replaces header from the MTA matches either the From or To header value of the active call at EP_T. This is checked by EP_T, and not by the proxy. Therefore, the CMS/Proxies merely pass the Dcs-Replaces header unchanged to EP_T.

7.8.1.2 CMS/Proxy Handling of Dcs-Also

7.8.1.2.1 Handling of Dcs-Also at $CMS/Proxy_I$

If Dcs-Also contains a private-param, then $CMS/Proxy_I$ MUST decrypt the information and extract the destination address and CMS_R address. The destination E.164 number and MTA address MUST be used to form a DCS-URL to address the requested destination.

If the Dcs-Also header had an attached Dcs-State header, CMS/Proxy$_I$ MUST decrypt it an extract the information needed in the following paragraphs. If the Dcs-Also header had no attached Dcs-State header, CMS/Proxy$_I$ MUST use the Dcs-State header for the existing call in progress.

CMS/Proxy$_I$ MUST check for an outstanding lawfully authorized surveillance order for the originating subscriber. If found, CMS/Proxy$_I$ MUST attach a Dcs-Laes header and a Dcs-Redirect header to the Dcs-Also. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content required, and MUST include a random string for use as a security key between the Delivery Functions. The Dcs-Redirect header MUST include the original destination, the forwarding subscriber, and the number of redirections that have occurred.

CMS/Proxy$_I$ MUST query the gate from the identified CMTS to obtain the call's billing information. CMS/Proxy$_I$ MUST copy the Dcs-Billing-Info and Dcs-Billing-ID from the original INVITE request and insert an additional Dcs-Billing-Info header to indicate that the requesting party will pay for the new call segment.

If the private-param or Dcs-State header value included Dcs-Anonymity information, an additional Dcs-Anonymity header MUST be included in the Dcs-Also header, carrying the endpoint's privacy request.

These additional headers MUST be combined into the Dcs-Also header, using the SIP syntax for optional headers attached to a SIP-URL (i.e. Dcs-Also: URL ? header=value & header=value).

7.8.1.2.2 Handling of Dcs-Also at CMS/Proxy$_T$

CMS/Proxy$_R$ MUST form a private-param for MTA$_R$, and include this private-param in the Dcs-Also header in the INVITE message passed to MTA$_T$. This private-param is formed from the following information: 1) the destination E.164 address and MTA address (if already known), 2) the value of Dcs-Billing-ID, 3) the sequence of Dcs-Billing-Info values, which indicate the complex charging arrangement for the new call, 4) an expiration time very shortly in the future, to limit the ability of MTA$_T$ to re-use this private-param for multiple calls, 5) the value of Dcs-Anonymity, and 6) the electronic surveillance information, if present.

Any remaining headers MUST be carried in the Dcs-Also header, using the SIP syntax for optional headers attached to a SIP-URL.

7.8.1.3 CMS/Proxy Handling of SDP

7.8.1.3.1 Handling of SDP at CMS/Proxy$_I$

CMS/Proxy$_I$ MUST extract the value of Dcs-Gate from the Dcs-State value.

On receipt of the 183-Session-Progress response, CMS/Proxy$_I$ MUST use the list of CODECs specified in the SDP payload to authorize maximum resources that can be used during the call at the initiating CMTS (CMTS$_I$). This information is used to signal the originating Gate Controller to send a Gate-Setup command to CMTS$_I$, defining the new envelope of the authorized QoS parameters. The Gate Controller MAY update the authorization at the pre-existing gate, or establish a new gate. If a new gate is established, the 183-Session-Progress response MUST include a Media-Authorization header giving the token for the resources.

If Dcs-Anonymity was present in the original call, and specified FULL or IPAddr, CMS/Proxy$_I$ MUST provide IP address privacy through the use of an anonymizer service. See Appendix Z for further details. CMS/Proxy$_I$ MUST modify the SDP connection information based on the IP address provided by the anonymizer service in the request that is sent to CMS$_R$.

7.8.1.3.2 Handling of SDP at CMS/Proxy$_R$

CMS/Proxy$_R$ MUST extract the value of Dcs-Gate from the Dcs-State value. CMS/Proxy$_R$ MUST use the list of CODECs specified in the SDP payload to signal the terminating Gate Controller. The Gate Controller determines whether the pre-existing gate can be re-used, or whether a new gate needs to be established. If a new gate is established, the INVITE request MUST include a Media-Authorization header giving the token for the resources.

On receipt of the 183-Session-Progress response, CMS/Proxy$_R$ MUST use the list of CODECs specified in the SDP payload to update the authorization at the gate.

If Dcs-Anonymity was present in the original call, and specified FULL or IPAddr, CMS/Proxy$_R$ MUST provide IP address privacy through the use of an anonymizer service. See Appendix Z for further details. CMS/Proxy$_R$ MUST modify the SDP connection information based on the IP address provided by the anonymizer service in the request that is sent to MTA$_R$.

7.8.2 CMS/Agent$_I$ Initiating Call Hold: INVITE(hold)

To place a call on hold, an INVITE(hold) message is sent on the end-end signaling channel to the endpoint that is to be put on hold. It is a standard SIP INVITE message, with the IP address in the connection field in SDP ("c=") set to 0.0.0.0. To maintain privacy of the initiator, this INVITE message SHOULD NOT contain a Dcs-Remote-Party-ID header, nor a Dcs-Anonymity header. The format of the INVITE message sent by the initiating endpoint (CMS/Agent$_I$) and the requirements on the header fields checked at the receiving endpoint (EP$_R$) are as follows:

INVITE(Hold): (CMS/Agent$_I$ -> EP$_R$) **Header:**	***Requirements on CMS$_I$ for message generation*** ***Requirements on CMS$_R$ for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header received for this call.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.* *MUST contain the IP address or FQDN of the initiating CMS.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the hold is initiated by the called party.*
To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 7.4.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST be set to 0.0.0.0*

On receiving an INVITE(hold), EP$_R$ MUST send the 200-OK with the updated SDP description to EP$_I$, and stop sending bearer channel packets to that same party.

200-OK (EP$_R$ -> CMS/Agent$_I$) **Header:**	***Requirement for EP$_R$ for message generation*** ***Requirement for CMS$_O$ for message checking***

SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(Hold) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(Hold).*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (...)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST be set to 0.0.0.0*

EP_I sends an ACK to EP_R. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

ACK: (CMS/Agent$_I$ -> EP$_R$) Header:	***Requirement at CMS$_I$ for message generation***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the most recent Contact header*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(Hold) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(Hold) request, method MUST indicate ACK*

After sending the INVITE(hold), the initiator MUST wait for a 200-OK response, then stop sending bearer channel packets and send an ACK to the other endpoint. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

7.8.3 CMS/Agent$_I$ Resuming a held call: INVITE(resume)

The endpoint that placed the call on hold MUST be the one to take it off hold. To take a call off hold, an INVITE(resume) is sent. An INVITE(resume) is an INVITE(hold) message with the SDP description of the call being reinstated. To maintain privacy of the initiator, this INVITE message SHOULD NOT contain a Dcs-Remote-Party-ID header, nor a Dcs-Anonymity header. The format of the INVITE message sent by the initiating endpoint (CMS/Agent$_I$) and the requirements on the header fields checked at the receiving endpoint (EP_R) are as follows:

INVITE(Resume): (CMS/Agent$_I$ -> EP$_R$) Header:	***Requirements on CMS$_I$ for message generation*** ***Requirements on CMS$_R$ for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header for this call.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.* *MUST contain the IP address or FQDN of the originating CMS.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the hold is initiated by the called party.*
To:	
Call-ID:	

CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 7.4.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST NOT be set to 0.0.0.0*

If EP_R is willing to reinstate the bearer channel, it MUST update the SDP description for the call and send a 200-OK with the updated SDP description to EP_I. If not, it MUST send a 4xx (client error) response. The 200-OK response MUST be as follows:

200-OK (EP_R -> CMS/$Agent_I$) **Header:**	***Requirement for EP_R for message generation*** ***Requirement for CMS_O for message checking***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(Resume) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(Resume).*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *The connection field (c=) MUST NOT be set to 0.0.0.0*

The endpoint that receives the 200-OK MUST send a standard SIP ACK message.

ACK: ($CMS/Agent_I$ -> EP_R) **Header:**	***Requirement at CMS_I for message generation***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the most recent Contact header*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(Resume) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(Resume) request, method MUST indicate ACK*

7.8.4 CMS/Agent$_R$ Receiving Call Hold: INVITE(hold) and INVITE(resume)

On receiving an INVITE(hold), CMS/Agent$_R$ MUST send the 200-OK with the updated SDP description to the party requesting the hold, and stop sending bearer channel packets to that same party. The expected response is an ACK.

The endpoint that was placed on hold MUST wait for an INVITE(resume), which is an INVITE(hold) message with the SDP description of the call being reinstated. If CMS/Agent$_R$ is willing to reinstate the bearer channel, it MUST update the SDP description for the call and send a 200-OK with the saved SDP description for the active call. If not, it MUST send a 4xx (client error) response. The expected response is an ACK.

See section 7.8.2 and 7.8.3 for description of the header fields in each message.

7.8.5 CMS/Agent$_I$ Initiating Blind Call Transfer

Two types of call transfer are described in this specification. Blind transfer is when the party initiating the transfer sequence does not have an active connection to the desired new destination. Therefore, the transferring party has no assurance that the call transfer will be successful. Consultative transfer is when the party initiating the transfer sequence has an active connection to the desired new destination. Both are realized with a combination of Dcs-Also and Dcs-Replaces headers in an INVITE request. This section describes only blind transfer; the next section describes consultative transfer.

By initiating a blind call transfer, the initiator is agreeing to be billed for a logical call-leg from himself to the new destination for the duration of the transferred call.

An INVITE request containing both a Dcs-Also header and a Dcs-Replaces header is used to transfer a call in progress. The “Dcs-Also:” header indicates the new party to be added and the “Dcs-Replaces:” header indicates the party to be removed. This INVITE is referred to as INVITE(also,replace).

The INVITE request for a blind transfer MUST be sent on the proxy-proxy signaling connection. The REQUEST URI MUST be the value of the most recently received Contact header.

The call-leg identification (From, To, and Call-ID) MUST match an active call. If the call originator is initiating the blind transfer, From and To will match those in the initial INVITE. If the call destination is initiating the blind transfer, the values of From and To will be reversed.

The Dcs-Also header MUST contain the Dcs-URL of the desired destination. This Dcs-URL MUST NOT contain a private-param. An additional header parameter (e.g. Dcs-Also: URL ? header=value & header=value) MUST be attached with “Dcs-Billing-ID=” and the value of the billing correlation value assigned to the original call. An additional header parameter MUST be attached with “Dcs-Billing-Info=” and the value of the billing information for the initial call from the original call. A third additional header parameter MUST be attached with “Dcs-Billing-Info=” and the accounting information for the second logical call leg from the initiator to the new destination.

CMS/Agent$_I$ MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber, or for surveillance active on the call to the initiating subscriber. If found, CMS/Agent$_I$ MUST include a “Dcs-Laes=” header parameter in the Dcs-Also. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call’s event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be delivered, and MUST include a random string for use as a security key between the Delivery Functions. CMS/Agent$_I$ MUST include a “Dcs-Redirect=” header parameter in the Dcs-Also. The Dcs-Redirect header MUST include the original destination, the forwarding subscriber, and the number of redirections.

Additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The Dcs-Replaces header MUST contain the same value as the From header, which refers to the initiating endpoint.

The INVITE request MUST include all the Dcs-State headers given to the CMS/Agent with matching call-leg identification (From, To, Call-ID).

The INVITE request MUST NOT contain an SDP description.

The requirements on the headers which CMS/Agent$_I$ MUST include in the message are shown below:

INVITE(also,replace): (CMS/Agent$_I$->CMS$_R$) Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*MUST be present. The Request-URI MUST be a DCS-URL as defined in Section 4.2.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present and MUST be the address of the originator of this message. Typically, the terminating CMS/Agent of the active call originates the INVITE(also,replace).*
Dcs-Also: DCS-URL ? Dcs-Billing-ID=xx & Dcs-Billing-Info=yy & Dcs-Billing-Info=zz	*MUST be present and identifies the new address of the destination to which the recipient of this INVITE(also,replace) is to issue an INVITE. Identifies new call leg to be created.* *MAY be any valid DCS-URL but MUST NOT contain a private-param. Attached header parameters MUST be as described above.*
Dcs-State:	*MUST be present if a Dcs-State header was present in a request or response received by CMS/Agent$_I$*
From:	*MUST be either the From or To header of the current call being transferred, whichever is the address of the originator of this message. Typically, INVITE(also,replace) is sent by the call-ed party, and therefore this is the To header.*
To:	*MUST be either the From or To header of the current call being transferred, whichever is the address of the destination of this message. Typically, this is the call-ing party, and therefore this is the From header*
Call-ID: ID	*The Call-ID MUST be the same as the Call-ID of the active call*
CSeq: n INVITE	*MUST be as defined in 7.4.* *Method MUST be INVITE*
Dcs-Replaces:	*This identifies the call-leg to be torn down at the endpoint receiving the INVITE(also,replace).* *It MUST be identical to the value of the From header in this message.*

A call flows illustrating the use of INVITE(also,replace) in blind call-transfer is shown in Appendix N.

The INVITE(also,replace) traverses through CMSs to the destination. Receipt of this message by a CMS/Agent is described in section 7.8.8, and by a CMS/Proxy in section 7.8.1.2.2.

7.8.6 CMS/Agent$_I$ Initiating Consultative Call Transfer

Both consultative and blind transfer are realized with a combination of Dcs-Also and Dcs-Replaces headers in an INVITE request. This section describes only consultative transfer; the previous section described blind transfer.

Consultative transfer is when the party initiating the transfer sequence (the initiator) has an active connection to the client (the client), and also has an active connection to the desired new destination (the consultant). Typically the client had previously called the initiator; then the initiator called the consultant and decided to transfer the client to the consultant.

By initiating a consultative call transfer, the initiator is agreeing to be billed for a logical call-leg from himself to the consultant for the duration of the transferred call. If the client had initially called the initiator, then the billing of the resulting transferred call will be split between the client and the initiator. If the initiator had initially called the client, then the billing of the resulting transferred call will be entirely to the initiator.

An INVITE request containing both a Dcs-Also header and a Dcs-Replaces header is used to transfer a call in progress. The "Dcs-Also:" header indicates the new party to be added and the "Dcs-Replaces:" header indicates the party to be removed. This INVITE is referred to as INVITE(also,replace).

The INVITE request for a consultative transfer MUST be sent on the proxy-proxy signaling connection, to be forwarded to the consultant. The REQUEST URI MUST be the value of the most recently received Contact header.

The call-leg identification (From, To, and Call-ID) MUST match the active call with the consultant. If the call originator is initiating the consultative transfer, From and To will match those in the initial INVITE. If the call destination is initiating the consultative transfer, the values of From and To will be reversed.

The Dcs-Also header MUST contain the Dcs-URL of the client, as received by the initiator in the Dcs-Remote-Party-ID header. This Dcs-URL MUST NOT contain a private-param. An additional header parameter (e.g. Dcs-Also: URL ? header=value & header=value) MUST be attached with "Call-ID=" and the value of the Call-ID for the existing call between the initiator and the client. An additional header parameter MUST be attached with "Dcs-Replaces=" and the value of either From or To of the call-leg identification (whichever refers to the initiator) of the call between the initiator and the client. If any state headers are present for the call between the initiator and the client, an additional header parameter MUST be attached with "Dcs-State=" and the value(s) of those state headers.

An additional header parameter MUST be attached with "Dcs-Billing-ID=" and the value of the billing correlation value assigned to the call between the initiator and the client. An additional header parameter MUST be attached with "Dcs-Billing-Info=" and the value of the billing information for the call between the initiator and the client. A third additional header parameter MUST be attached with "Dcs-Billing-Info=" and the accounting information for the second logical call leg from the initiator to the consultant.

CMS/Agent$_I$ MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber, or for surveillance active on the call to the initiating subscriber. If found, CMS/Agent$_I$ MUST include a "Dcs-Laes=" header parameter in the Dcs-Also. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be delivered, and MUST include a random string for use as a security key between the Delivery Functions. CMS/Agent$_I$ MUST include a "Dcs-Redirect=" header parameter in the Dcs-Also. The Dcs-Redirect header MUST include the original destination, the forwarding subscriber, and the number of redirections.

Other additional header parameters SHOULD NOT be attached to the Dcs-Also header.

The Dcs-Replaces header MUST contain the same value as the From header, which refers to the initiating endpoint.

The INVITE request MUST include the Dcs-State header (if any) given to CMS/Agent$_I$ with matching call-leg identification (From, To, Call-ID).

The INVITE request MUST NOT contain an SDP description.

The requirements on the headers which the CMS/Agent MUST include in the message are shown below:

INVITE(also,replace): (CMS/Agent$_I$->CMS$_R$) Header:	***Requirement on CMS$_I$ for message generation***
INVITE SIP-URL SIP/2.0	*MUST be present.* *The Request-URI MUST be a SIP-URL as defined in Section 4.2.* *Request-URI MUST be from the most recent Contact header from Initial INVITE or initial 183 the active call with the consultant.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present and MUST be the address of the originator of this message. Typically, the terminating CMS/Agent of the active call originates the INVITE(also,replace).*
Dcs-Also: DCS-URL ? Call-ID=ID & Dcs-Replaces=URL & Dcs-Billing-ID=xx & Dcs-Billing-Info=yy & Dcs-Billing-Info=zz	*MUST be present and identifies the client.* *MAY be any valid DCS-URL.* *MUST be copied from the Dcs-Remote-Party-ID of the call with client.* *MAY include a private-param in a DCS-URL.* *Call-ID, Dcs-Replaces, Dcs-Billing-ID, and Dcs-Billing-Info attached headers MUST be present, and be as described above.*
Dcs-State:	*MUST be present if a Dcs-State header was present in a request or response received by CMS/Agent$_I$*
From:	*MUST be either the From or To header of the call with consultant, whichever is the address of the originator of this message. Typically, consultative transfer is sent by the calling party, and therefore this is the From header.*
To:	*MUST be either the From or To header of the call with consultant, whichever is the address of the destination of this message. Typically, this is the call-ed party, and therefore this is the To header*
Call-ID: ID	*The Call-ID MUST be the same as the Call-ID of the active call with consulotant*
CSeq: n_i INVITE	*MUST be as defined in 7.4.* *Method MUST be INVITE*
Dcs-Replaces:	*MUST be identical to the value of the From header in this message.*

A call flow illustrating the use of INVITE(also,replace) in consultative call-transfer is shown in Appendix O.

The INVITE(also,replace) traverses through CMSs to the destination. Receipt of this message by a CMS/Agent is described in section 7.8.8, and by a CMS/Proxy in section 7.8.1.2.2.

7.8.7 CMS/Agent$_I$ Initiating an Ad-hoc Conference

An ad-hoc conference is formed when an initiator has two simultaneous active calls, one to party A and one to party B, and desires to connect them together. While it is possible to do this locally, within the endpoint, it consumes double the access network resources of a conference bridge and is therefore discouraged. The CMS/Agent SHOULD NOT direct the endpoint to perform local bridging of multiple calls over the cable access network.

When creating a call with multiple parties connected to a single destination, e.g. to a bridge for an ad-hoc conference, it is often more convenient and efficient to request the destination to initiate the additional calls, rather than initiate a call transfer to direct each party to the desired new destination.

An INVITE request containing two or more Dcs-Also headers is used to initiate an ad-hoc conference. The "Dcs-Also" header indicates the party to be added to the conference. This INVITE is referred to as INVITE(also).

Ad-hoc conference initiation involves establishing a new connection from the conference initiator to the bridge service. This new connection has all the properties of a normal call, and the INVITE message MUST contain all the header fields as described in 7.6.1. The INVITE request for an ad-hoc conference MUST be sent over the proxy-proxy signaling path, to be forwarded to the bridge service.

A CMS/Agent that sends INVITE(Also) is indicating a willingness to pay for the additional call segments between the endpoint and the bridge for all of the parties in the conference. The CMS/Agent establishes proper billing arrangements.

The URI of a bridge server MUST be provisioned in CMS/Agent$_I$. The Request-URI of the INVITE MUST be this provisioned URI.

The INVITE(Also) message sent by CMS/Agent$_I$ to initiate an ad-hoc conference MUST be as follows:

INVITE (also): (CMS/Agent$_I$ -> CMS$_R$): Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*The Request URI MUST be a DCS-URL, as specified above*
--all other headers--	*All other headers are unchanged from INVITE as in 7.6.1*
Dcs-Also: DCS-URL ? Call-ID=ID-A & Dcs-Replaces=URL-A	*See requirements below.*
Dcs-Also: DCS-URL ? Call-ID=ID-B & Dcs-Replaces=URL-B	*See requirements below.*
	An empty line MUST be present between the headers and the message body
--SDP description--	*MUST be an SDP description, as specified in 7.6.1*

The INVITE(also) request initiating an ad-hoc conference MUST contain two or more Dcs-Also headers, one header for each participant in the conference.

For each participant in the ad-hoc conference (referred to as party-X in this paragraph), the Dcs-Also: header MUST contain the DCS-URL obtained from the Dcs-Remote-Party-ID header of the INVITE message or 183-Session-Progress for the active call with party-X. This MUST NOT be a DCS-URL containing a private-param. An additional header parameter (e.g. Dcs-Also: URL ? header=value & header=value) MUST be attached with "Call-ID=" and the value of the Call-ID for the existing call between the initiator and party-X. An additional header parameter MUST be attached with "Dcs-Replaces=" and the value of either From or To of the call-leg identification (whichever refers to the initiator) of the call between the initiator and party-X. If any state headers are present for the call between the initiator and party-X, an additional header parameter MUST be attached with "Dcs-State=" and the value(s) of those state headers.

An additional header parameter MUST be attached with "Dcs-Billing-ID=" and the value of the billing correlation value assigned to the call between the initiator and party-X. An additional header parameter MUST be attached with "Dcs-Billing-Info=" and the value of the billing information for the call between the initiator and party-X. A third additional header parameter MUST be attached with "Dcs-Billing-Info=" and the accounting information for the second logical call leg from the initiator to the bridge.

CMS/Agent$_I$ MUST check for an outstanding lawfully authorized surveillance order for the initiating subscriber, or for surveillance active on the call between party-X and the initiating subscriber. If found, CMS/Agent$_I$ MUST include a "Dcs-Laes=" header parameter in the Dcs-Also. The Dcs-Laes header MUST include the address and port of the local Electronic Surveillance Delivery Function for a copy of the call's event messages, MUST include the address and port of the local Electronic Surveillance Delivery Function for the copy of call content if call content is to be delivered, and MUST include a random string for use as a security key between the Delivery Functions. CMS/Agent$_I$ MUST include a "Dcs-Redirect=" header parameter in the Dcs-Also. The Dcs-Redirect header MUST include the original destination, the forwarding subscriber, and the number of redirections.

Other additional header parameters SHOULD NOT be attached to the Dcs-Also header.

A call flow illustrating the use of INVITE(also) in establishing an ad-hoc conference is shown in Appendix P.

7.8.8 Call Control: CMS/Agent$_R$ Receiving INVITE(also/replace)

This section describes the handling of INVITE(Also), INVITE(Replace), and INVITE(Also,replace), collectively referred to here as INVITE(also/replace). When both Dcs-Also headers and Dcs-Replaces headers are present in the same INVITE, the Dcs-Also headers are processed first, followed by the Dcs-Replaces headers.

Typical use of INVITE(also/replace) is for call features such as call-transfer and three-way-calling, where the other party in the call initiated the special feature and is requesting the receiving CMS/Agent to alter its current connections in support of that feature.

A CMS/Agent MUST be capable of receiving an INVITE(also/replace) message from its peer CMSs at any time during an active call. The INVITE(also/replace) message received at CMS/Agent$_R$ is shown in the table below.

INVITE(also/replace) (CMS$_I$-> CMS/Agent$_R$): Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*MUST be present. MUST be sufficient for the CMS/Agent to determine the proper endpoint being addressed.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.*
Dcs-Also: DCS-URL	*If present, this header contains a DCS-URL, and MAY contain attached headers.*
From:	*Call identification {From, To, CallID}. MAY refer to either an existing call or a new session.*
To:	
Call-ID:	
CSeq:	*MUST be present. Method MUST be INVITE.*
Dcs-Replaces:	*If present, this header contains a SIP-URL, and MAY contain attached headers.* *MUST match either the From: header or To: header of a current call.* *Call-ID (or a Call-ID attached header) MUST match same current call.*

The INVITE(also/replace) MAY contain an SDP description, and, if so, indicates a new media session MUST be established before processing the Dcs-Also or Dcs-Replaces headers. Procedures for establishing the media session are identical to Section 7.6.

CMS/Agent$_R$ MUST check the validity of the Dcs-Also and Dcs-Replaces headers. A CMS/Agent support an endpoint that is not capable of performing local bridging of media streams SHOULD reject an INVITE(also/replace) that would result in two or more call legs with the same Call-ID once all the Dcs-Also and Dcs-Replaces headers have been processed. For each Dcs-Repalces header, CMS/Agent$_R$ MUST verify it has a call active with the matching Call-ID and that the Dcs-Replaces value matches either the From or To call-leg identification

Once any media session is established, and CMS/Agent$_R$ has checked the validity of the Dcs-Also and Dcs-Replaces headers, CMS/Agent$_R$ MUST send the final response to the INVITE(also/replace). The final response to an INVITE(also/replace) is a 200-OK as in an INVITE.

On receiving an INVITE(also/replace) that includes the Dcs-Also: header, CMS/Agent$_R$ MUST send an INVITE message for each Dcs-Also header present. The Request-URI MUST be the DCS-URL from the Dcs-Also header of the received INVITE(also/replace), and the To: header MUST be filled in by CMS/Agent$_R$ as a locally unique string. The To: header is therefore an identifier without any significance to the caller or the final destination of the INVITE. Any additional headers given with the DCS-URL in the Dcs-Also: header MUST be copied into the INVITE.

INVITE (CMS/Agent$_R$ ->CMS$_X$) Header:	***Requirement***

INVITE DCS-URL SIP/2.0	*The Request-URI is the DCS-URL from the Dcs-Also header in the received INVITE(Also/Replace).*
To: sip:participant1	*Contents of the To: header MUST be filled in by the CMS/Agent as a locally unique string. MUST be different from the From: header.*
--all headers appearing in Dcs-Also: as additional headers--	*MUST be copied from the additional header component of the DCS-URL in the Dcs-Also header of the received INVITE(Also/Replace).*
--all other headers--	*All other headers are unchanged from 7.6.1*
	An empty line MUST be present between the headers and the message body
--SDP description--	*MUST be an SDP description, as specified in 7.6.1*

Procedures to complete the setup of the session with media are identical to the basic call described in 7.6.

On receiving an INVITE(also/replace) that includes the Dcs-Replaces: header, CMS/Agent$_R$ MUST verify it has a call active with the matching Call-ID and that the Dcs-Replaces value matches either the From or To call-leg identification. If all is proper, CMS/Agent$_R$ MUST send a BYE message to the party identified in the Dcs-Replaces header, and terminate that call.

7.8.9 Operator Services: Initiating INVITE(BLV) and INVITE(EI)

Operator Services (Busy Line verification and Emergency Interrupt) are initiated from the CMS/Agent on behalf of a PSTN gateway connecting to special MF trunks groups from the OSPS system. The SIP messages INVITE(BLV) and INVITE(EI) are initiated by CMS/Agent$_O$. These messages include the Dcs-OSPS header. An INVITE(BLV) has Dcs-OSPS set to BLV.

The INVITE(BLV) message sent by CMS/Agent$_O$ to initiate a busy line verification MUST be as follows:

INVITE (BLV): (CMS/Agent$_O$ -> CMS$_T$): Header:	***Requirement***
INVITE DCS-URL SIP/2.0	*The Request URI MUST be a DCS-URL, as in 7.6.1*
Dcs-OSPS: BLV	*MUST be present*
Dcs-Remote-Party-ID: [display-name] <DCS-URL>;rpi-type=operator	*MUST be present. MUST contain "rpi-type=operator"*
--all other headers--	*All other headers are unchanged from INVITE as in 7.6.1*
	An empty line MUST be present between the headers and the message body
--SDP description--	*MUST be an SDP description, as specified in 7.6.1*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmissions MUST stop on receipt of any response.

The remainder of the call establishment, from the view of the originator, proceeds identically to that of a basic call given in 7.6.

On receipt of an indication from the Media Gateway that an intercept tone is present on the line, CMS/Agent$_O$ initiates an INVITE(EI) to convert the call to an emergency interrupt session. The INVITE(EI) is sent over the end-end signaling path, as follows:

INVITE(EI): (CMS/Agent$_O$ -> EP$_T$) Header:	***Requirement***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header received.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.* *MUST contain the IP address or FQDN of the originating CMS.*
From:	*MUST be present. MUST be same as INVITE(BLV)*

To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 7.4.*
Dcs-OSPS: EI	*MUST be present*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of any response. The expected response is a 200-OK, as follows:

200-OK (CMS_T -> CMS/$Agent_O$) **Header:**	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITEIEI) message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(EI) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(EI).*

On receipt of the 200-OK, CMS/$Agent_O$ MUST respond with an ACK message.

ACK: (CMS/$Agent_O$ -> CMS_T) **Header:**	***Requirement***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value in the most recent Contact header.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(EI) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(EI) request, method MUST indicate ACK*

7.8.10 Operator Services: Receipt of INVITE(BLV) and INVITE(EI)

Operator Services (Busy Line verification) and Emergency Interrupt are initiated from the CMS/Agent on behalf of a PSTN gateway connecting to special MF trunks groups from the OSPS system. The SIP messages INVITE(BLV) and INVITE(EI) are initiated by the CMS/Agent. These messages include the Dcs-OSPS header. An INVITE(BLV) has Dcs-OSPS set to BLV.

CMS/$Agent_T$ MUST be prepared to receive an INVITE(BLV) at any time. If not received over an IPSec-secured link from another CMS, it SHOULD be rejected. It SHOULD NOT result in a busy error response. It MUST NOT result in alerting the user. If the Dcs-Remote-Party-ID header does not contain a rpi-type of "Operator," CMS/$Agent_T$ SHOULD reject the message.

Invite(BLV): (CMS_O->CMS_T) **Header:**	***Requirement***
INVITE sip:E.164-t | host-t@Host(cms-t); user=phone | ip SIP/2.0	*MUST be present. Identifies the line that is to be verified as busy.*
Dcs-Remote-Party-ID:User-o <tel:E.164-o>;rpi-type=operator	*MUST be present. MUST contain Caller-Type of "Operator."*
Dcs-Osps: BLV	*MUST be present. MUST be set to BLV.*
--all other headers, including SDP---	*MUST be as specified for INVITE*

CMS/Agent$_T$ MUST respond to INVITE(BLV) with a 183-Session-Progres, and the call completes as in Sections 7.6.2.2, 7.6.4, and 7.6.7.

The SDP describes the media flow from the endpoint to the PSTN gateway; CMS/Agent$_T$ SHOULD cause a packet stream to be sent to that address. The endpoint could perform a mixing operation between the two ends of an active call, and send the mixed stream to the OSPS system. The endpoint could check for voice activity locally, and if none send a copy of the received voice stream. The endpoint could send a duplicate copy of the locally-generated voice stream.

If the telephone line is idle, CMS/Agent$_T$ SHOULD cause a stream of silence packets to be sent to the OSPS system. If the telephone line is ringing, or locally generating a ringback tone, CMS/Agent$_T$ SHOULD cause a ringback sequence to be sent to the OSPS system.

The operator may decide to interrupt the call after confirming that the line is busy, and signals this intention by placing an alerting tone on the voice path to the endpoint. The MG at the PSTN Gateway detects this tone and the CMS/Agent for that MG formulates an INVITE(EI) message. This message is a variant of the INVITE with Dcs-OSPS header set to EI. This INVITE(EI) message is sent over the end-end signaling channel from CMS$_O$ to CMS/Agent$_T$.

CMS/Agent$_T$ MUST be prepared to accept an INVITE(EI) at any time a BLV call is active. The INVITE(EI) is defined in the following table:

INVITE(EI): (CMS$_O$ -> CMS/Agent$_T$) Header:	***Requirement***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header received.*
From:	*From:, To:, and Call-ID: MUST be present. MUST be a direct copy of the corresponding headers from the INVITE(BLV) message sent previously.*
To:	
Call-ID:	
CSeq: n+1 INVITE	*MUST be present.* *Call sequence number "n" MUST be one greater than previous request message. Method MUST be INVITE.*
Dcs-Osps: EI	*MUST be present. MUST be equal to EI.*

If CMS/Agent$_T$ receives INVITE(EI) but has not previously received INVITE(BLV) with identical call-leg identification, it MUST reject the message.

CMS/Agent$_T$ responds to INVITE(EI) with a 200-OK final response.

200-OK: (CMS/Agent$_T$ -> CMS$_O$) Header:	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITEIEI) message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE(EI) message.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE(EI).*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-response. The default value of (T- proxy-response) is given in Appendix A. Retransmissions MUST stop on receipt of the following ACK message.

ACK: (CMS_O -> $CMS/Agent_T$) Header:	***Requirement***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value in the most recent Contact header received.*
Via:	*MUST be present.*
From:	*MUST be present.* *MUST be copies of same headers in INVITE(EI) request.*
To:	
Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(EI) request, method MUST indicate ACK*

On acceptance of a valid INVITE(EI), $CMS/Agent_T$ MUST enable communication between the operator and the local user. $CMS/Agent_T$ MAY place the existing call on hold and switch to the operator call (e.g. call-waiting). Alternatively, if resources are available, $CMS/Agent_T$ could establish a three-way call with the operator and the current party or parties.

7.8.11 SIP Messages for CODEC Changes – INVITE(Codec-change)

The INVITE(Codec-change) message is sent by either endpoint to initiate a change in the codec. There are two separate cases described. First is a change to a codec that was in the original set of codecs listed in the initial INVITE request. Resource authorization has already been performed, and the message exchange occurs only between the endpoints to synchronize the change.

The second case is a change to a coded that was not previously specified in the initial INVITE. The Gate Controller component of the CMS need to be involved in this to increase the resource authorization, and therefore the message exchange goes along the proxy-proxy signaling path.

7.8.11.1 Codec Change within previous authorization

If the new codec that $CMS/Agent_I$ wishes to change to was included in the SDP of the initial INVITE transaction (or authorized by a subsequent INVITE(Codec-Change) request), the codec is considered authorized by the network.

In this case, $CMS/Agent_I$ initiating the codec change MUST send an INVITE message directly to the other endpoint with the new codec description. To maintain privacy of the initiator, this INVITE message SHOULD NOT contain a Dcs-Remote-Party-ID header, nor a Dcs-Anonymity header. The format of the INVITE message sent by the initiating CMS/Agent ($CMS/Agent_I$) and the requirements on the header fields checked at the receiving CMS/Agent (EP_R) are as follows.

INVITE(Codec-change): ($CMS/Agent_I$ -> EP_R) Header:	***Requirements on EP_I for message generation*** ***Requirements on EP_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header received.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.* *MUST contain the IP address or FQDN of the originating CMS/Agent.*
From:	*MUST be present. MUST be same as initial INVITE for the call being placed on hold, but with From: and To: reversed if the change is initiated by the called party.*
To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 7.4.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present*

	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *MUST include "a=X-pc-csuite"'s and "a=X-pc-secret" with the previous ciphersuites and keying material, indicating no change is desired.* *MUST contain line "a=X-pc-qos: mandatory sendrecv"*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmission MUST stop on receipt of a provisional or final response.

On receiving an INVITE(Codec-change), CMS/$Agent_R$ MUST match it to the existing call by the use of the From, To, and Call-ID headers. If there is no match, CMS/$Agent_R$ considers this a new call attempt from a non-PacketCable endpoint, and MAY ignore it. CMS/$Agent_R$ MUST send a 183-Session-Progress provisional response, giving the agreed codec.

183-Session-Progress: (EP_R -> CMS/$Agent_I$) Header:	***Requirement for EP_R for message generation*** ***Requirement for EP_I for message checking***
SIP/2.0 183 Session Progress	*Status line with status code 183 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the INVITE message.* *Identifies the call.*
To:	
Call-ID:	
Contact:	*MUST be present. MUST be same as in 183-Session-Progress*
CSeq:	*MUST be present. MUST be the same as in the INVITE.*
RSeq: x_t	*MUST be present.*
Session: qos	*MUST be present, and MUST contain value "qos"*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (...)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *MUST include "a=X-pc-csuite"'s and "a=X-pc-secret" with the previous ciphersuites and keying material, indicating no change is desired.* *MUST contain "a=X-pc-qos: sendrecv mandatory confirm"*

The retransmission timer for this message SHOULD be set to T-direct-response. The default value of (T-direct-response) is given in Appendix A. Retransmissions MUST stop on receipt of PRACK.

CMS/$Agent_I$ MUST send a PRACK to acknowledge receipt of the 183-Session-Progress. The PRACK message MUST be sent directly to the address specified in the most recent Contact header.

An SDP MUST be included in the PRACK message. The SDP in the PRACK MUST include a media (m=) line with a single CODEC to be used for this connection.

PRACK: (CMS/$Agent_I$ -> EP_R) Header:	***Requirement at EP_I for message generation***
PRACK SIP-URL SIP/2.0	*MUST be present. Method MUST be PRACK. The value of the SIP-URL MUST be the most recent Contact header received*

Via:	*MUST be present. MUST be the IP address or FQDN of EP_I.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in the provisional response.*
Call-ID:	
Cseq: n_o+1 ACK	*Sequence number 'n_o+1' MUST be one higher than previous sequence number, method MUST indicate PRACK*
Rack: x n_o INVITE	*Value 'x' MUST be a copy of the value in the Rseq header of the 183-$_o$' MUST be a copy of the Cseq value from the INVITE request. Method MUST be INVITE.*
Content-Type: application/sdp	*MUST be present, and MUST be as defined in 4.6.4..*
Content-length: (…)	*MUST be present.*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be present* *Contains the SDP description as modified after processing the SDP returned by the terminating endpoint, and MUST contain a single CODEC choice.*

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A. Retransmissions MUST stop on receipt of 200-OK.

$CMS/Agent_I$ MUST instruct the endpoint to reserve the resources required. $CMS/Agent_I$ sends a PRECONDITION-MET message, or other failure message, to EP_R. This is as shown in 7.6.3.

$CMS/Agent_R$ MUST send a 200-OK acknowledgement to the PRACK (as in section 7.6.4), and use the SDP description in the PRACK message to instruct the endpoint to reserve access network resources. If successful and after receiving a PRECONDITION-MET message from EP_I, CMS/AgentR MUST send a 200-OK acknowledgement to the PRECONDITION-MET (as in section 7.6.4) and sends a 200-OK final response to the INVITE(codec-change) to EP_I.

On sending the 200-OK, $CMS/Agent_R$ instructs the endpoint to commit the network resources. It MAY start sending using the new codec.

200-OK: (EP_R -> $CMS/Agent_I$) **Header:**	***Requirement***
SIP/2.0 200 OK	*Status line with status code 200 MUST be present.*
Via:	*MUST be present, copy from INVITE message.*
Dcs-State:	*MUST be present, copy of value from INVITE message*
From:	*From:, To: and Call-ID MUST be present and MUST be copied from the received INVITE.* *Identifies the call.*
To:	
Call-ID:	
CSeq:	*MUST be present. MUST be the same as in the INVITE.*

On receiving a 200-OK response, $CMS/Agent_I$ instructs the endpoint to commit network resources and MAY start using the new codec. $CMS/Agent_I$ MUST send out an ACK directly to EP_R. The ACK follows the rules for an ACK sent in response to 200-OK for an INVITE message.

ACK: ($CMS/Agent_I$ -> EP_R) **Header:**	***Requirement at MTA_I for message generation***
ACK SIP-URL SIP/2.0	*MUST be present. Method MUST be ACK. SIP-URL MUST be the value received in the most recent Contact header received.*
Via:	*MUST be present.*
From:	*MUST be present.*
To:	*MUST be copies of same headers in INVITE(Codec-Change) request.*

Call-ID:	
Cseq: n_i ACK	*Sequence number MUST be copy of CSEQ value in INVITE(CodecChange) request, method MUST indicate ACK*

Example call flows for CODEC change within previous authorization are included in Appendix R.

7.8.11.2 Codec Change requiring new authorization

If the codec desired by EP_I or $CMS/Agent_I$ was not included in the SDP of the initial INVITE, and was therefore not authorized by the network, $CMS/Agent_I$ sends the INVITE(codec-change) request over the proxy-proxy signaling channel. This message MUST have the same To:, From:, and Call-ID: headers that identify the active call. The message traverses proxies like an INVITE message.

The format of the INVITE message sent by $CMS/Agent_I$ and the requirements on the header fields checked at the receiving CMS (CMS_R) are as follows.

INVITE(Codec-change): ($CMS/Agent_I$->CMS_R) Header:	***Requirements on CMS_I for message generation Requirements on CMS_R for message checking***
INVITE SIP-URL SIP/2.0	*Request line MUST be present.* *The request method MUST be set to INVITE.* *The Request URI MUST be the value of the most recent Contact header received for this call.*
Via: SIP/2.0/UDP Host(cms-i)	*MUST be present.* *MUST contain the IP address or FQDN of the originating CMS.*
Dcs-State:	*MUST be present if a previous request or response sent to CMS_I contained a Dcs-State header.*
From:	*MUST be present. MUST be same as initial INVITE for the call being modified, but with From: and To: reversed if the change is initiated by the called party.*
To:	
Call-ID:	
CSeq: n_i INVITE	*MUST be present.* *Call sequence number "n_i" MUST be as defined in 6.1.*
Content-Type: application/sdp	*MUST be present. MUST be as defined in 4.6.4..*
Content-length: (...)	*MUST be present*
	An empty line (CRCR, LFLF, or CRLFCRLF) MUST be present between the headers and the message body.
v= o= s= c= b= t= a= m=	*MUST be a SDP description as described in Section 5.* *MUST include "a=X-pc-csuite"'s and "a=X-pc-secret" with the previous ciphersuites and keying material, indicating no change is desired.* *MUST contain line "a=X-pc-qos: mandatory sendrecv"*

The retransmission timer (T1) for this message SHOULD be set to T-proxy-request. The default value of (T-proxy-request) is given in Appendix A. Retransmission MUST stop on receipt of a final response.

The remainder of the procedure for changing CODECs is identical to that described in Section 7.8.11.1, for a mid-call CODEC change that did not require an authorization change.

Example call flows for CODEC change requiring new authorization are included in Appendix S.

7.9 CMS handling of Call Teardown

To terminate a call, the CMS/Agent MUST send a BYE message on the end-end signaling channel and stop transmitting bearer data to the other endpoint. It MUST release network resources used for the call.

The retransmission timer (T1) for this message SHOULD be set to T-direct-request. The default value of (T-direct-request) is given in Appendix A.

We denote the endpoint that has detected local hangup by $CMS/Agent_I$; the other endpoint in the call is EP_R;

BYE: ($CMS/Agent_I$ -> EP_R) Header:	***Requirement***
BYE SIP-URL SIP/2.0	*Request line MUST include the BYE Method followed by the Contact header of the destination*
From:	*From, To, Call ID MUST be present to identify the call leg to be torn down. This is copied from the initial INVITE for this call, except that the From and To may be reversed if the termination is requested by the called party*
To:	
Call-ID:	
CSeq:	*The Sequence number MUST be as described in 7.4 Method MUST be BYE.*

Upon receipt of the BYE message, $CMS/Agent_R$ MUST release network resources that have been used for this call, and sends the following 200-OK message in response.

200-OK: (EP_R -> $CMS/Agent_I$) Header:	***Requirement***
SIP/2.0 200 OK	*Status line MUST include status code 200.*
From:	*From, To, Call-ID MUST be present and MUST be copied from the preceding BYE request.*
To:	
Call-ID:	
CSeq:	*MUST be present. Same as in the preceding BYE.*

Upon receipt of 200-OK, $CMS/Agent_I$ MUST stop the retransmission timer

8. Application Layer Anonymizer

Privacy of calling parties demands that the IP address of the caller not be known to the destination endpoint, since the IP address basically gives away caller-id information in spite of any request to keep it private. Likewise, IP addresses of the destination may not be revealed to the caller, in order to maintain privacy of transfer destinations.

In this Section we present an anonymizer as depicted below:

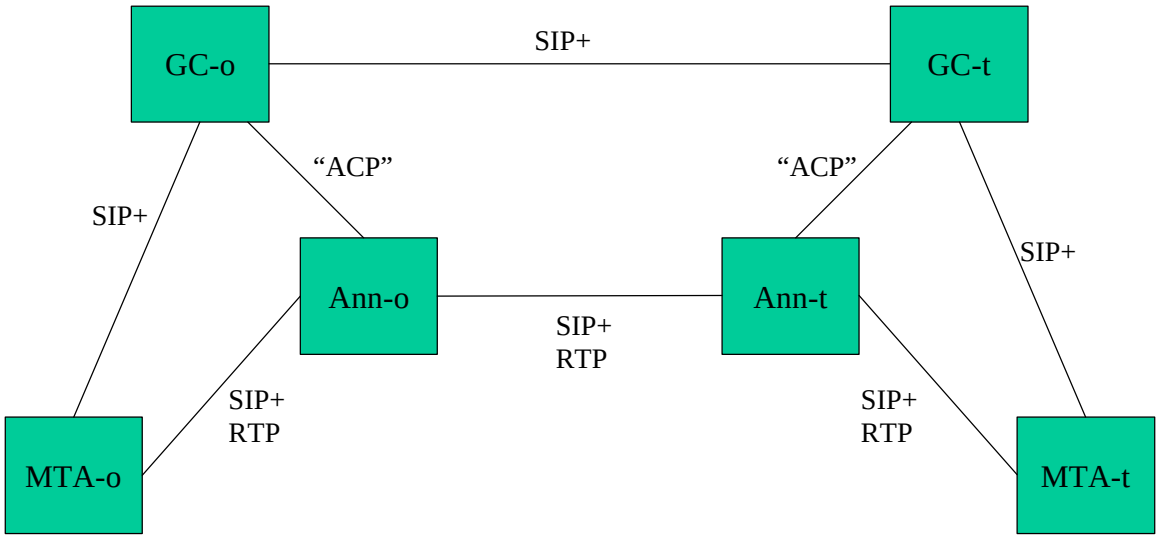


Figure 10: Anonymizer Interfaces

The initial SIP+ message exchange occurs through the CMS/Proxies. During this setup, the CMS/Proxies communicate with the anonymizer through some Anonymizer Control Protocol (ACP) to setup anonymizer sessions for the call. The subsequent SIP+ message exchange occurs through the anonymizers, and once the call is setup, the bearer channel is communicated through the anonymizer as well.

The second anonymizer we consider is an application level anonymizer. The application level anonymizer performs IP-address and port mappings just like the transport level anonymizer, however it also inspects the DCS/SIP+ messages and performs any modifications necessary to support the anonymizer session rather than relying on the MTA to perform this function.

The remainder of this section contains a call flow that illustrates how the above described transport level anonymizer can be supported.

The primary differences between the transport level anonymizer flow and this application level anonymizer flow are:

- ◊ Endpoints do not actively participate in the anonymizer service and are thus not provided with any anonymizer address information.
- ◊ The anonymizer inspects all call signaling messages and modify address information contained in them as needed.

8.1 Anonymizer Overview

This section is still work in progress, and is not considered normative at this time.

TBD

8.2 Anonymizer handling of Media

This section is still work in progress, and is not considered normative at this time.

TBD

8.3 Anonymizer Handling of SIP Messages

This section is still work in progress, and is not considered normative at this time.

TBD

8.4 Interface between Anonymizer and CMS

This section is still work in progress, and is not considered normative at this time.

TBD

9. SDL Description of MTA

This section is still work in progress, is not accurate, and is not considered normative at this time.

This section explains the mechanics associated with managing instantiations of the User Agent SIP MTA state machinery. This information is an aid for understand how the User Agent state machinery supports basic calls and error scenarios, as well as how multi-session features such as Call Waiting, 3-way calling, etc will be supported. The term's session and state machine instance are used interchangeably throughout this document and are equivalent.

9.1 Session Identification

Request-URI received at UAS MUST match a locally recognizable identifier (phone number, IP address & port number, or host name) that can be associated with a unique user interface telephone line.

The To header MUST carry the digits entered by the originating party.

If a To or From header is encrypted, then the characters that compose the encrypted string MUST remain constant for the life of the call leg.

The line identified by a Request-URI MAY have multiple simultaneously active state machine instances, and the MTA must be capable of supporting at least two simultaneously active state machine instances for each user interface telephone line.

Re-invites by a callee (such as in INVITE-replace) MAY have the To and From header fields reversed.

Events processed by a state machine may come from either the user interface or the MTA interface, the state machine (or session) MUST be identifiable by user interface telephone line and Request-URI.

A Call Leg is defined in http://www.cs.columbia.edu/~hgs/sip/notes.html as:

"the combination of local-address, remote-address, and call-id, where these addresses include tags. Only the username and host part of the To and From headers are used for this purpose, so that

"User 1" <sip:user1@columbia.edu>

and

"J.Doe" <sip:user1@columbia.edu>

would be the same when determining the call leg. [The] From and To headers designate the originator of the request, not that of the call leg."

9.2 Session Creation

[Intent is to define static and dynamic sessions]

The MTA MAY create a new instance of the state machine in response to the off hook user event if no state machine instance exists for the telephone line associated with the off hook event. Additionally, the MTA MAY create a new instance of the state machine in response to SIP requests and responses if no state machine instance exists for the associated Call Leg.

An instance of the state machine MUST exist for each Call Leg or an instance of the state machine MUST be dynamically created.

If a session is found to exist when an INVITE request is received from a different party, a 'parallel' sessions MUST be created to handle any simultaneous transaction processing necessary to convey BUSY status, call forwarding information, or support the Call Waiting feature.

9.3 Event Dispatching

User events MUST be dispatched to all active state machines associated with the line on which the event was received.

Requests MUST be delivered to the session identified by the associated Call Leg.

Responses MUST be delivered to the session identified by the associated Call Leg.

Responses that do not match any existing Call Legs MUST be ignored, and an error MAY be logged.

9.4 Event Filtering and Error Handling

The MTA MUST ignore SIP requests and responses with errors in the start-line, To header, From header, or Call-ID header. The MTA MUST ignore SIP requests and responses with unparsable the start-line, To header, From header, or Call-ID header.

9.5 MTA State Transition Diagram Overview

Shown in Figure 11 is an overview of the MTA state machine which shows the protocol control for basic calls.

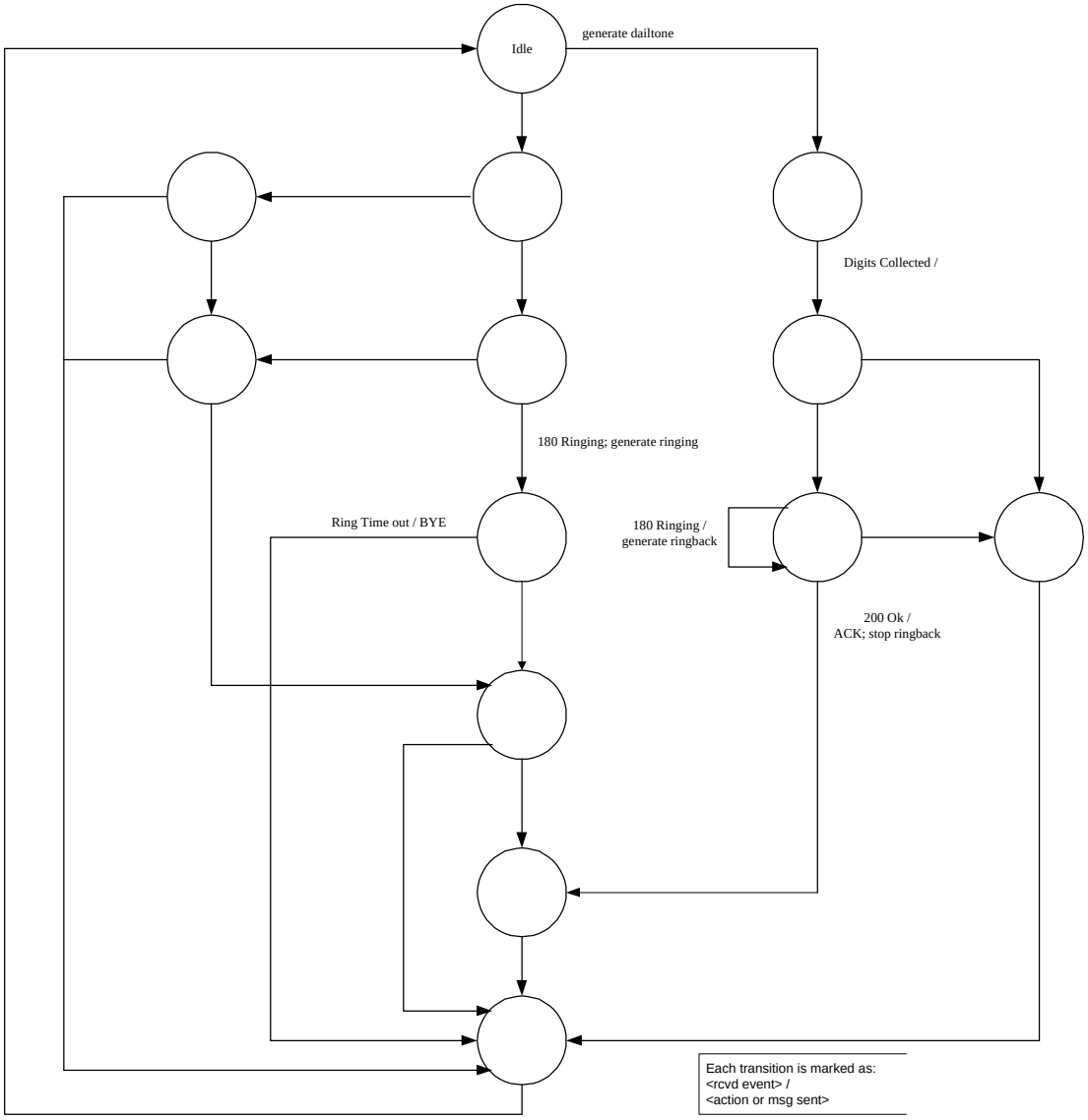


Figure 11: MTA State Transition Diagram Overview

9.6 MTA Transitions from Idle State

This state is entered when the call session is instantiated.

In response to the Invite request in the idle state, the MTA MUST parse the Invite request and verify all mandatory informational content as define in section 2.x.x (reference to MTA Interface) while in the idle state. If the MTA is unable to parse the Invite request or the Invite request does not contain the mandatory Invite (stage1) content:

1) the MTA MAY proved vendor specific unexpected event processing
2) the MTA MUST transmit a 4xx, 5xx, or 6xx final response and
3) the MTA MUST retransmit the final response in accordance with RFC 2543[11] (for illustration, the SDL's reset the retransmission count and start T1)

and the MTA enters the failure state.

In response to the user going off hook in the idle state, an MTA providing standard telephone interface

1) MUST provide the user with the dial tone
2) MUST start digit collection
3) MUST start T4 representing an off hook timeout

and enter the digit collect sate. An MTA not providing a standard telephone interface MAY use an alternative method to obtain the To header information.

[Other text to follow]

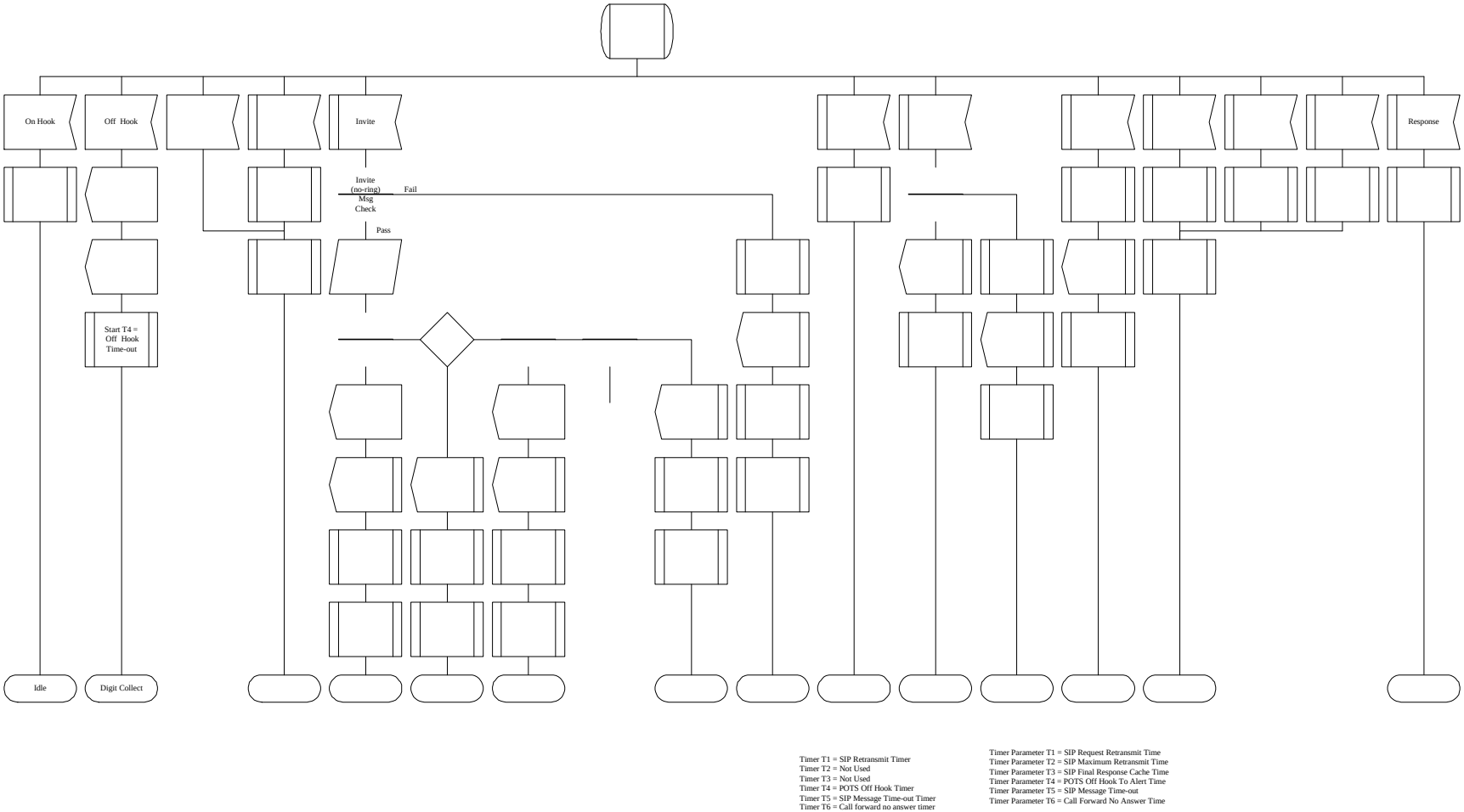


Figure 12: MTA Transitions from Idle State

9.7 MTA Transitions from Digit-Collect State

Text to be provided.

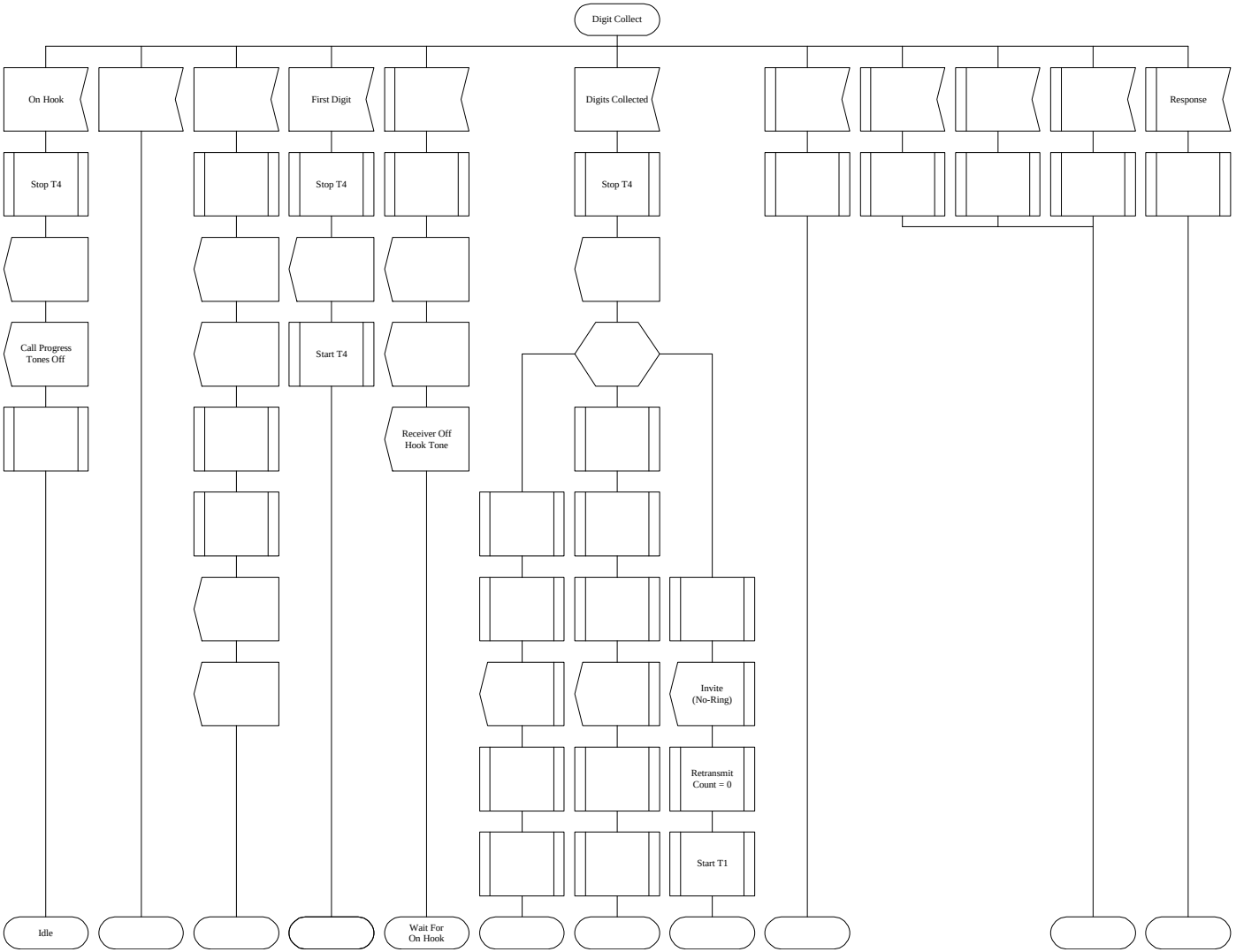


Figure 13: MTA Transitions from Digit-Collect State

9.8 MTA Transitions from Originating-Stage1 State

Text to be provided.

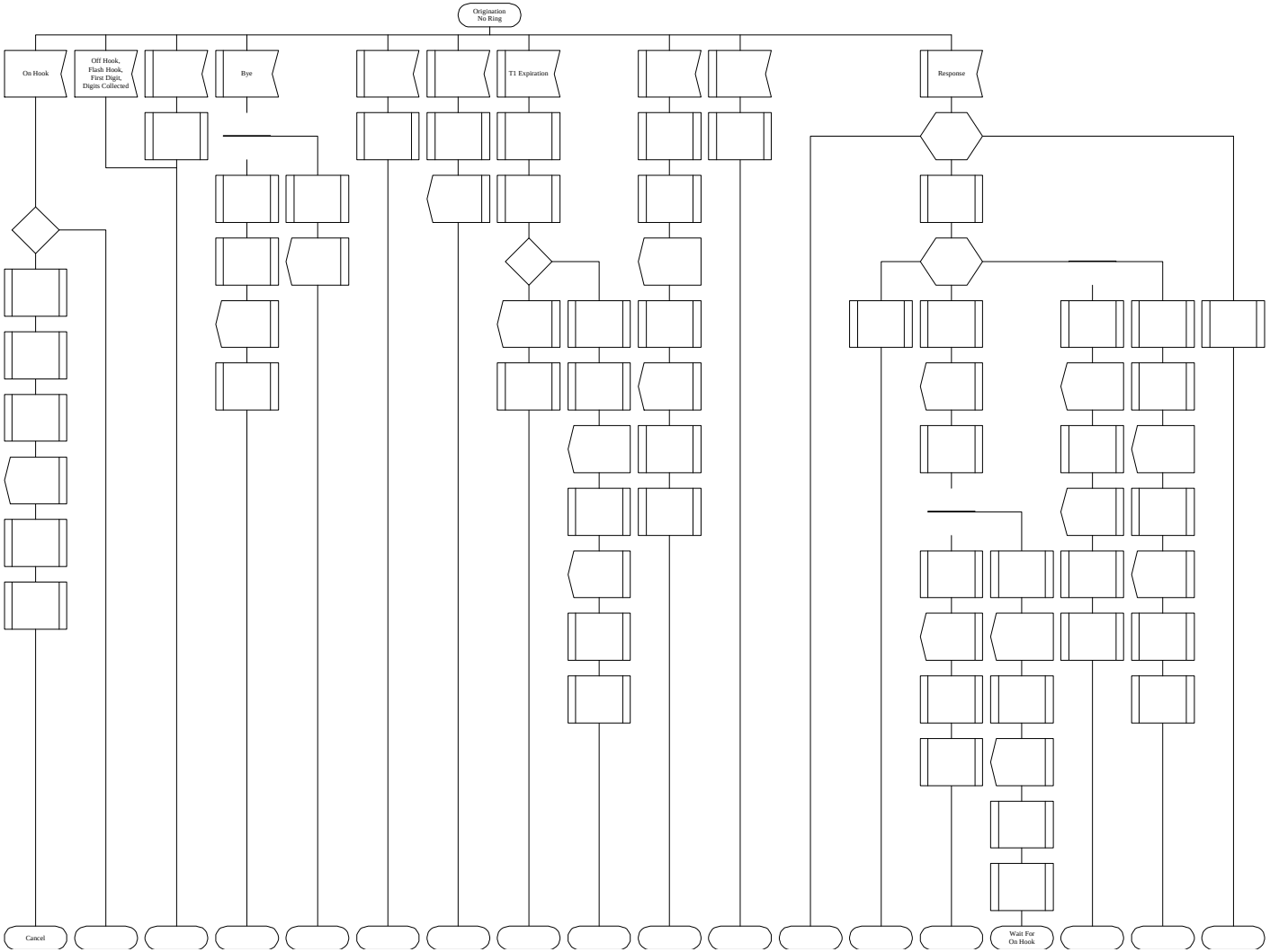


Figure 14: MTA Transitions from Originating-Stage1 State

9.9 MTA Transitions from Originating-Ring-Request State

Text to be provided.

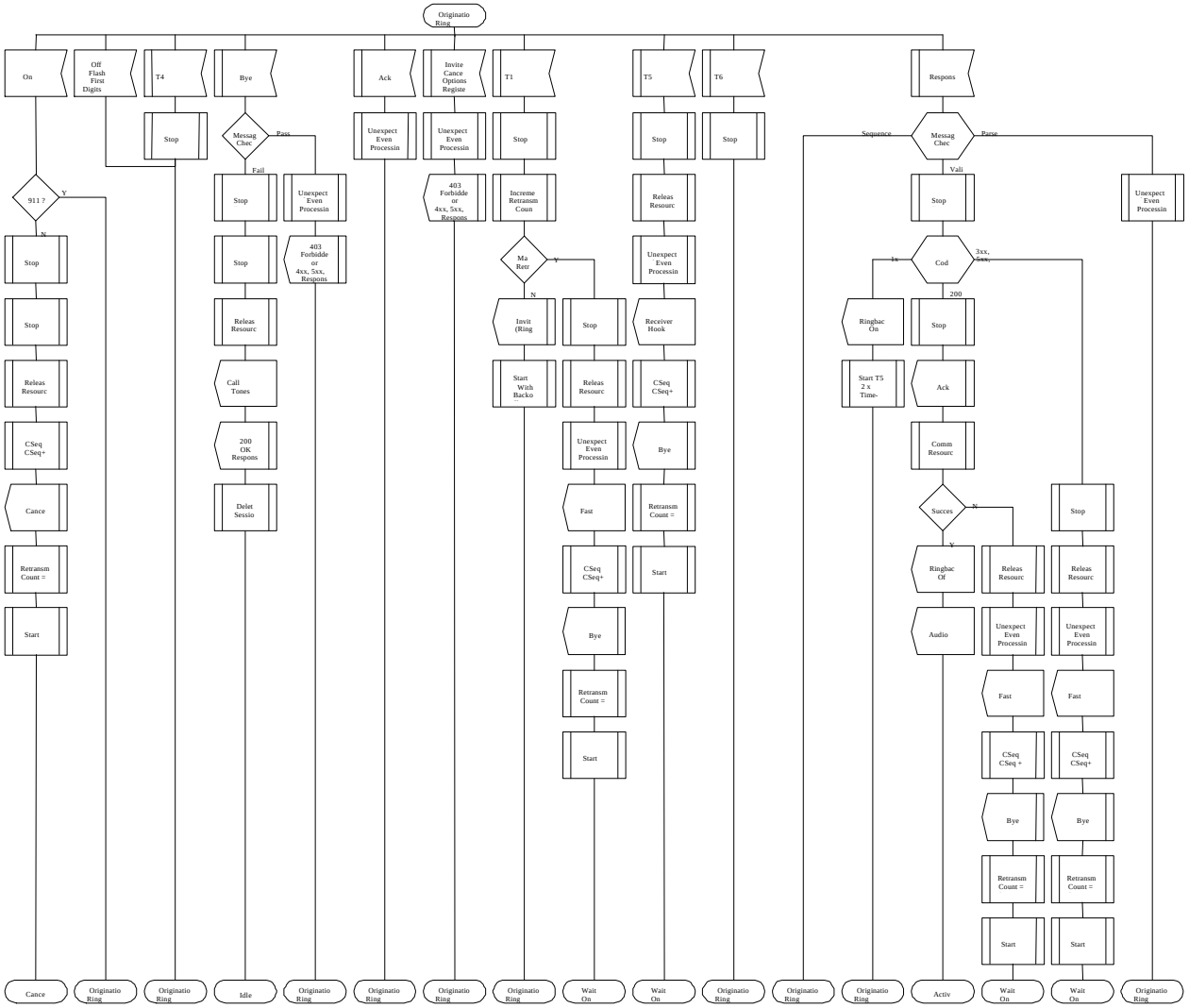


Figure 15: MTA Transitions from Originating-Ring-Request State

9.10 MTA Transitions from Terminating-Stage1 State

Entry Criteria:
This state is entered from the **IDLE** state upon reception of a valid "INVITE"
request with a Dcs-Stage1 header present. A 200-OK Response has been sent with the Callee's SDP description of the accepted capabilities.

Timers Running:
Retransmission timer T1 is running.

Expected (success) events - ACK:
The MTA MUST parse the ACK request and verify all mandatory informational content as define in section 2.x.x (MTA Interface reference). If the MTA is unable to successfully parse the ACK request or the ACK is missing mandatory informational content, the MTA MAY provide vender specific error processing and MUST ignore the ACK request.

The MTA MUST verify the ACK request CSeq is equal Invite (stage1) request CSeq. If the ACK request CSeq does not equal the Invite (stage1) request CSeq, the MTA MUST ignore the ACK request.

If the MTA determines the ACK request is valid, the MTA MUST stop the response retransmission timer T1 and stop retransmission of the 200-OK response. Additionally, the MTA MUST initiate bandwidth resource reservations based on the ACK request SDP contents (nice place for reservation requirement based on SDP). If the MTA successfully reserve resources, the MTA MUST start timer T5, transitions and to the Termination Contacted state. However, if the MTA is unable to reserve resources, the MTA MAY provide vender specific error processing, MUST set the CSeq to zero, MUST begin transmitting the BYE request as defined in section 2.x.x (reference to MTA interface), MUST set the retransmission count to zero, MUST start the retransmission timer T1and MUST transition state to Teardown.

the verify the ACK has a call sequence number Resource reservation will be attempted based on the information in the received SDP description. If resource reservation is successful, Timer T5 will be started to wait for the "INVITE-RING" request, and will cause a transition to the **Terminating Contacted** state. If resource reservation is unsuccessful, an error will be logged, a "BYE" request will be sent directly to the Caller using a Request-URI based on the Contact header received in the "INVITE" request. The To and From headers of the "BYE" request will be the reverse of the From and To headers in the "INVITE" request. Additionally, the retransmission timer T1 is started, the retransmission counter is reset and the statemachine transitions to the **Teardown** state.

Event Handling - Offhook:
If the user goes offhook in this state, glare occurs causing a statemachine transition to the **Terminating Glare Stage1** state. Note that the user will NOT receive dialtone. As soon as the second INVITE-RING transaction completes, the voice path will be cut through to the Calling party.

Event Handling - Invite:
An INVITE request may be received in this state that is either due to a retransmission (INVITE(Stage1)), or may be an INVITE-RING request which means that the ACK request was lost or has not arrived yet. If the CSEQ header matches a previously received value, (do we need to check the SDP too?) the INVITE is a duplicate. In this case, a 200-OK response is sent hop-by-hop through CMS/$Proxy_T$. The retransmission counter is reset, and retransmission timer T1 is started. No transition.

If the request is determined to be a valid INVITE-RING with an incremented CSEQ, T1 is stopped.. Resource reservation will be attempted based on the information in the received SDP description. If resource reservation is successful, turn power ringing on. If Caller ID is activated, and Caller ID information is present, pass the Caller ID information to the Caller ID device. Send a 180 Ringing

Response directly to the Caller using a Request-URI based on the Contact header received in the "INVITE" request. Start timer T6. Transition to the **Termination Ringing** state.

Event Handling - BYE:
A BYE may be received if the calling party has hung up. Stop T1. Send a 200-OK Response. Delete this session. Transition to the **IDLE** state. **Q: how can you transition to the IDLE state if the session is deleted?**

Event Handling - Cancel:
A CANCEL may be received if the calling party has hung up. Stop T1. Send a 200-OK Response. Start Retransmission timer T1, reset retransmission counter. Transition to the **Failure** state.

Event Handling - T1 Expiration:
Increment Retry count. Test to see if maximum retries exceeded. If exceeded, an error will be logged, a "BYE" request will be sent directly to the Caller using a Request-URI based on the Contact header received in the "INVITE" request. The To and From headers of the "BYE" request will be the reverse of the From and To headers in the "INVITE" request. Start retransmission timer T1, reset retransmission counter. Transition to the **Teardown** state.

Unexpected Events - Onhook, Flashook, First Digit, Last Digit:
Log unexpected event. Ignore. No transition.

Unexpected Events - T4 Expiration, T5 Expiration, T6 Expiration:
Stop timer. Log unexpected event. Ignore. No transition.

Unexpected Events – Options, Register:
Stop timer T1. Log unexpected event. Send 403 Forbidden (or 4xx, 5xx, 6xx?) response hop-by-hop via CMS/$Proxy_T$. Start retransmission timer T1, reset retransmission counter. Transition to the **Failure** state.

Unexpected Events – Response:
Stop timer T1. Log unexpected event. Send a "BYE" request hop-by-hop? How do we address the BYE? There's no Request-URI or Contact header in the response. I suggest we just pitch it. Start retransmission timer T1, reset retransmission counter. Transition to the **Teardown** state.

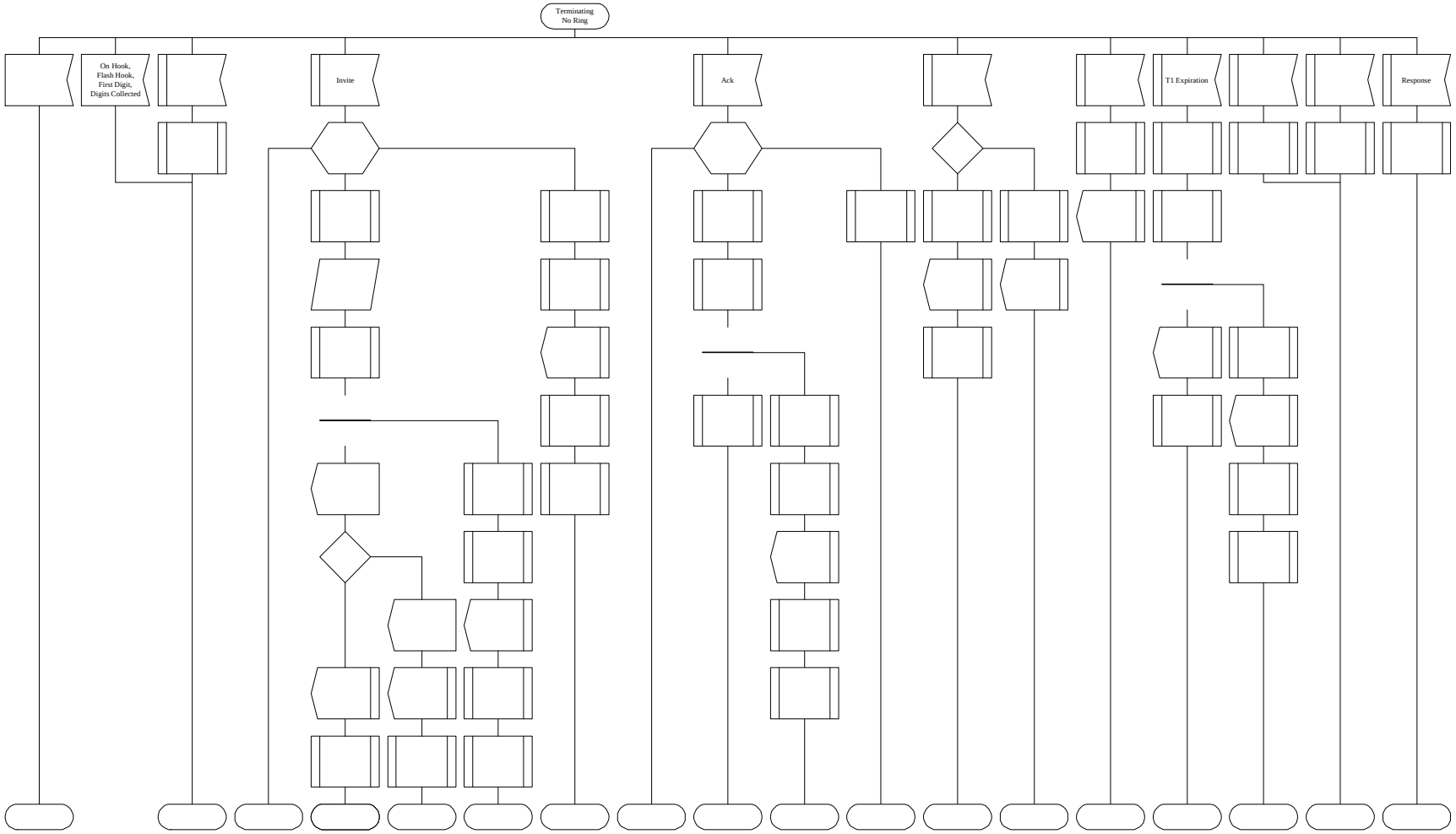


Figure 16: MTA Transitions from Terminating-Stage1 State

9.11 MTA Transitions from Terminating-Contacted State

Text to be provided.

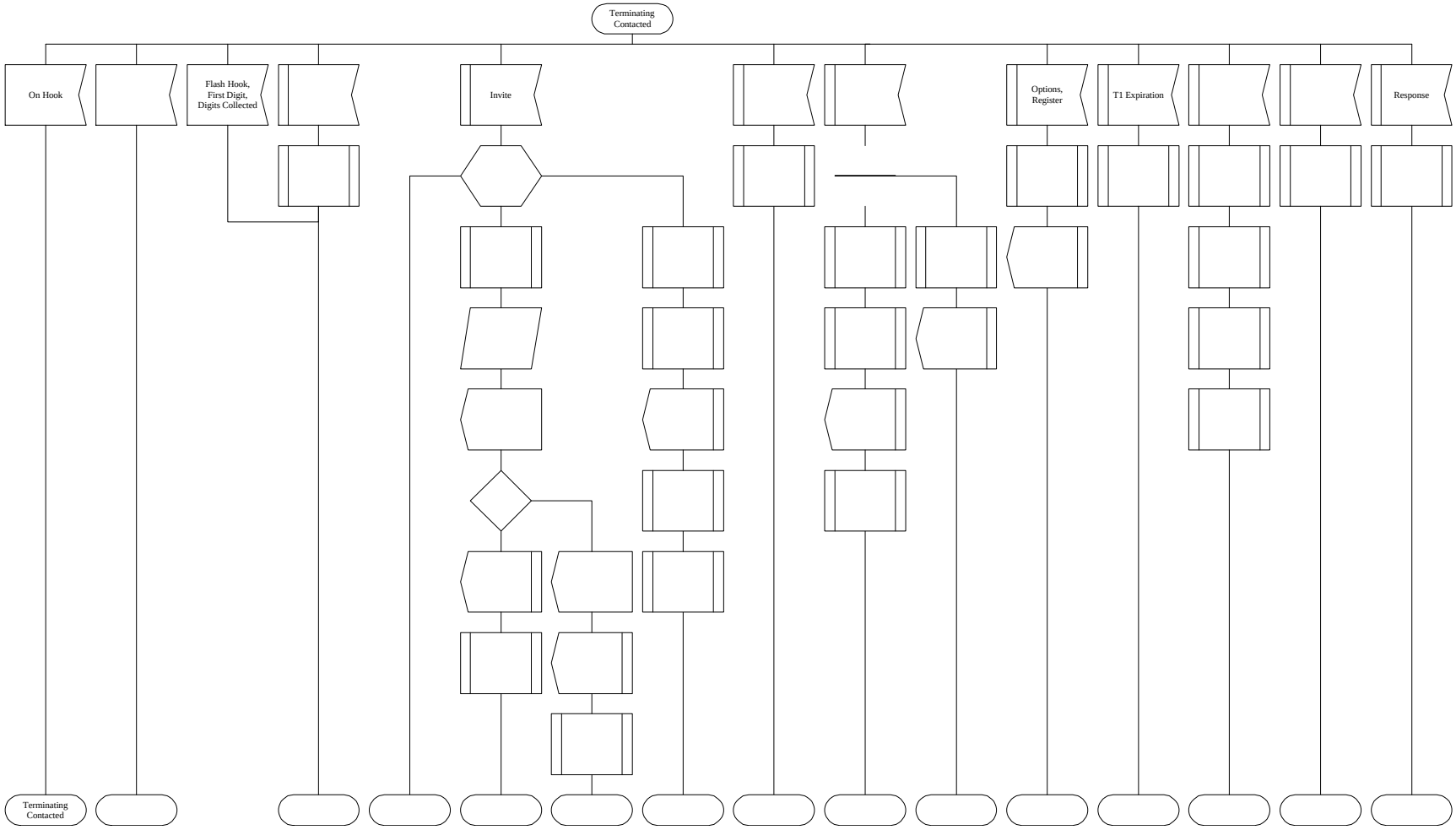


Figure 17: Terminating-Contacted State

9.12 MTA Transitions from Terminating-Ringing State

Text to be provided

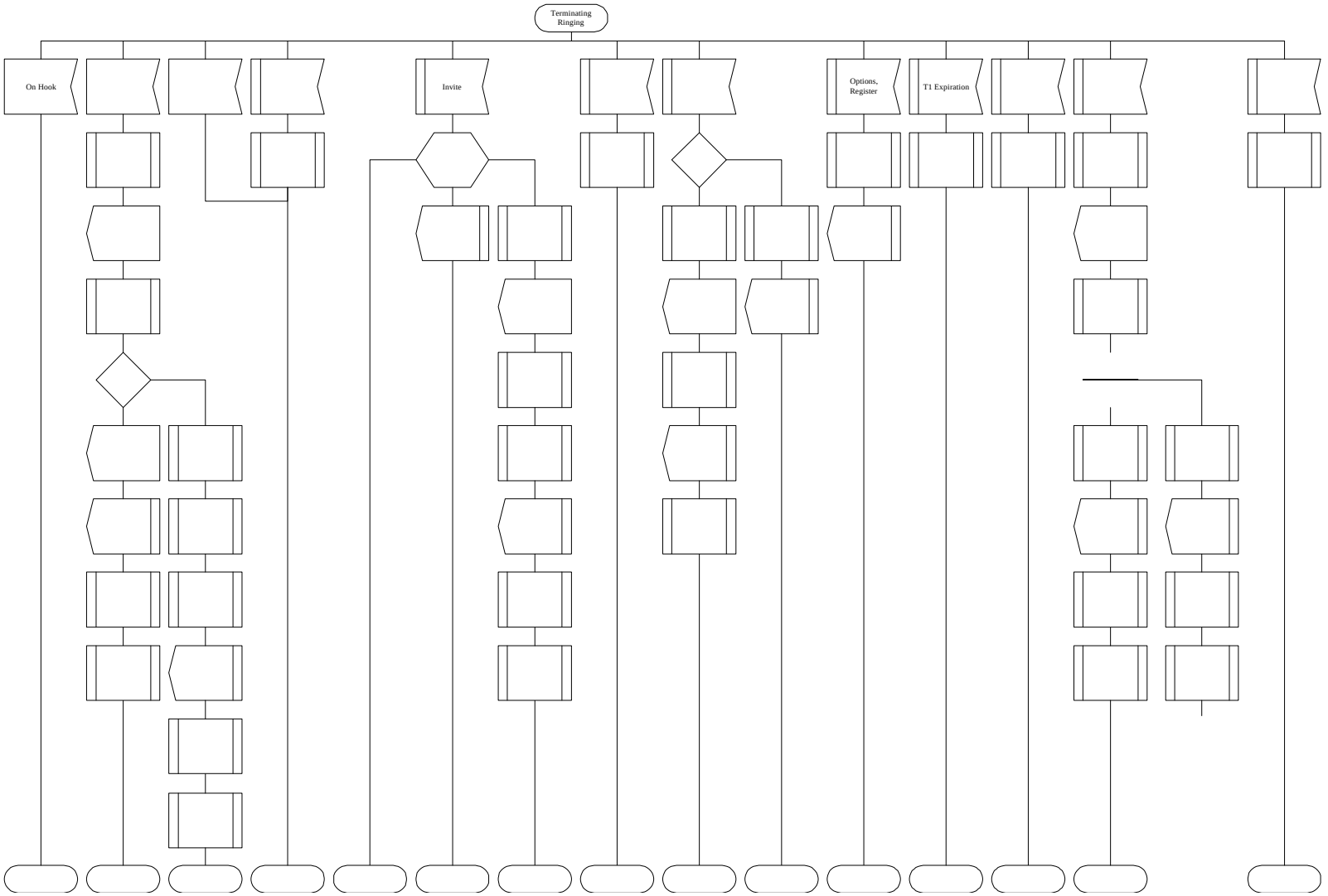


Figure 18: MTA Transitions from Terminating-Ringing State

9.13 MTA Transitions from Terminating-Glare-Stage1 State

Text to be provided

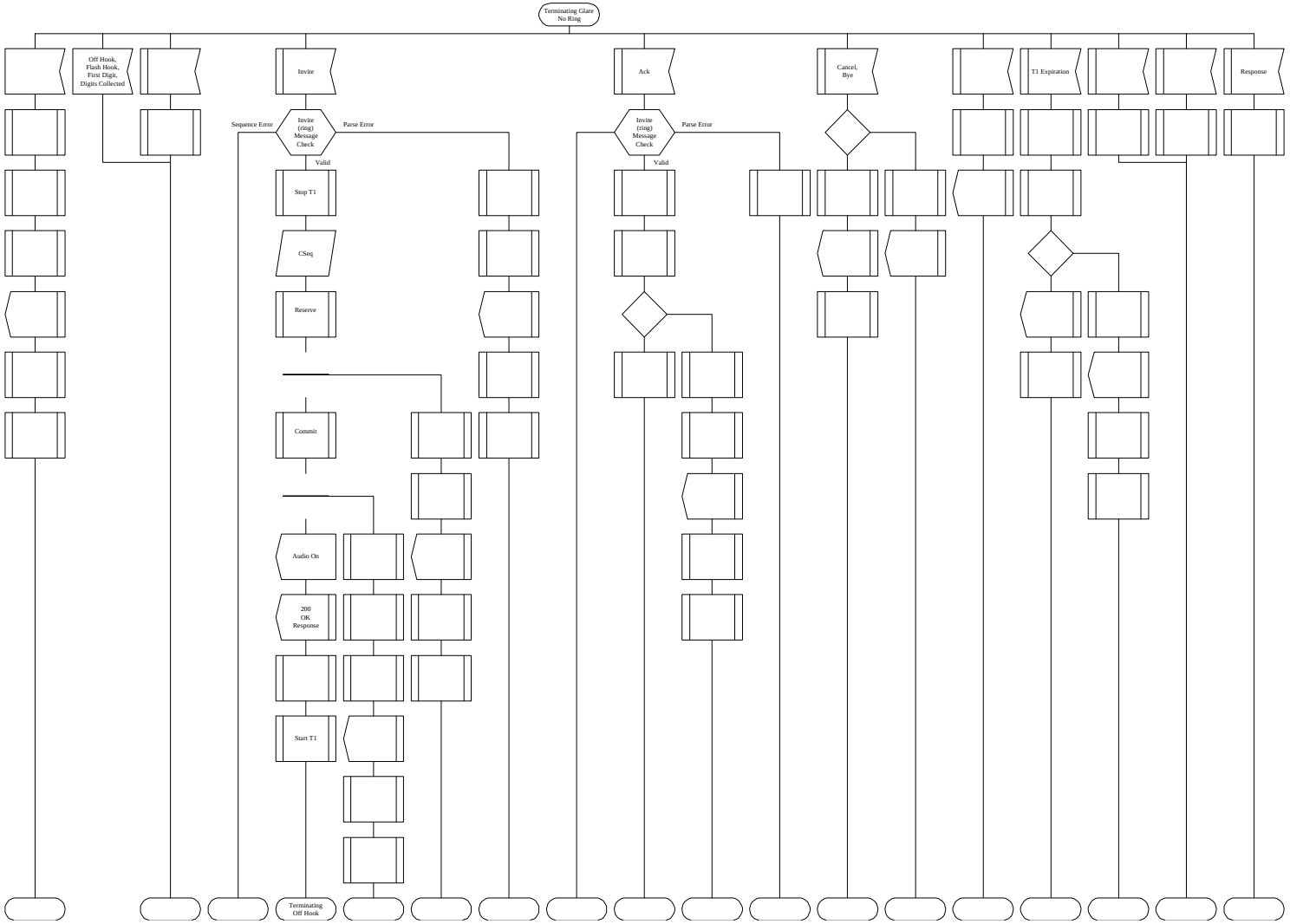


Figure 19: MTA Transitions from Terminating-Glare-Stage1 State

9.14 MTA Transitions from Terminating-Glare Contacted State

Text to be provided

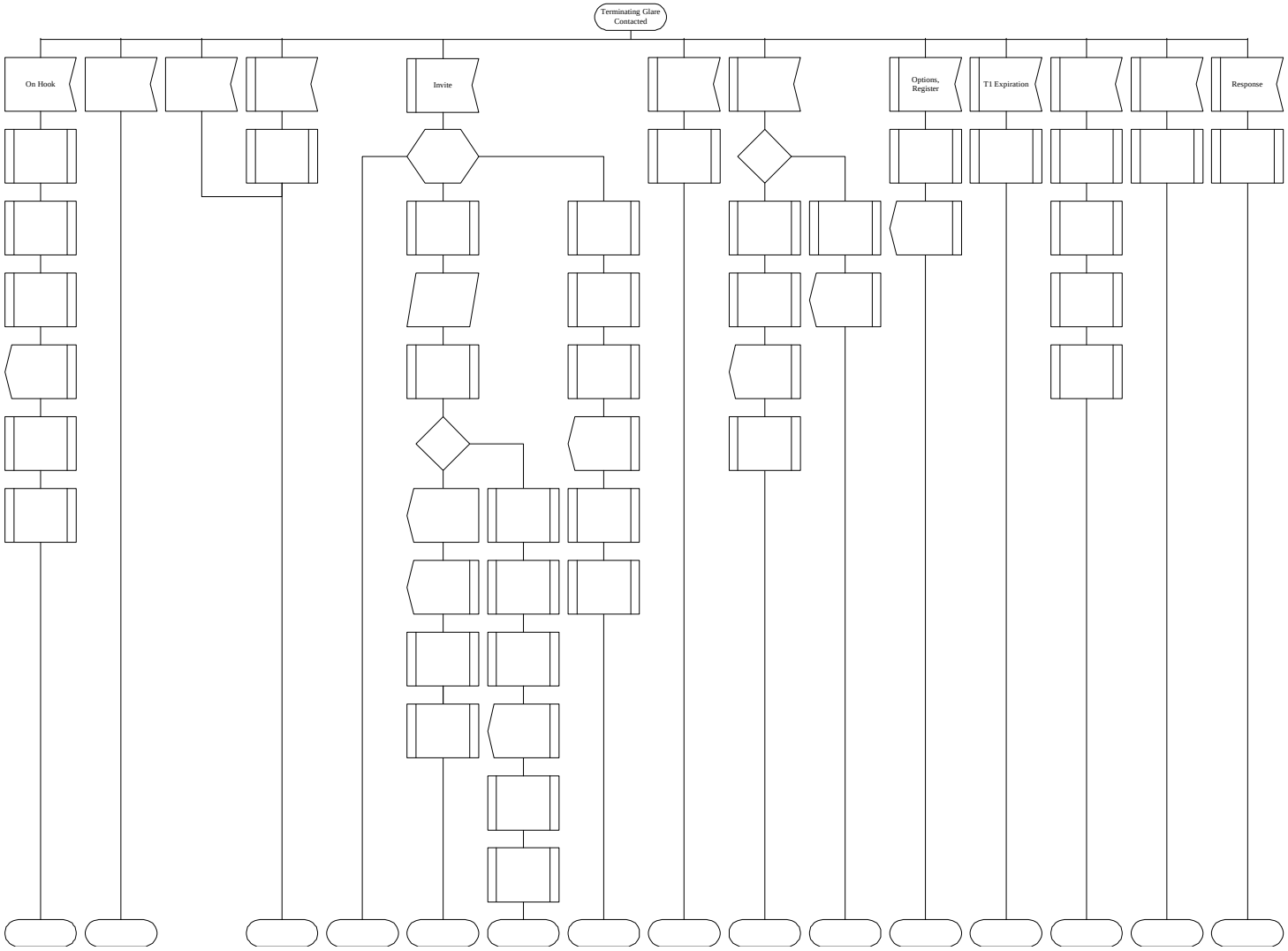


Figure 20: MTA Transitions from Terminating-Glare Contacted State

9.15 MTA Transitions from Terminating-Offhook State

Text to be provided

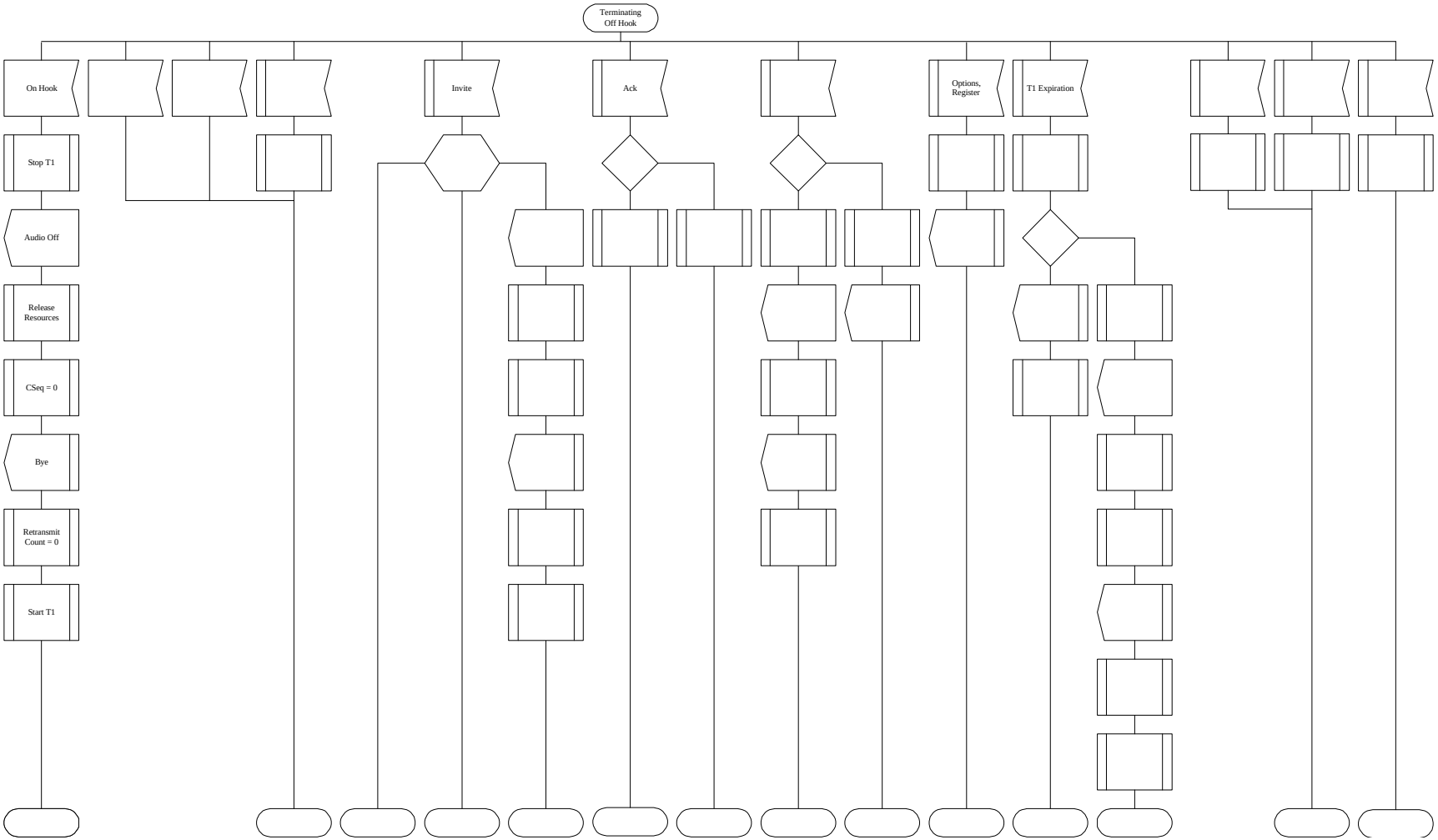


Figure 21: MTA Transitions from Terminating-Offhook State

9.16 MTA Transitions from Active State

Text to be provided

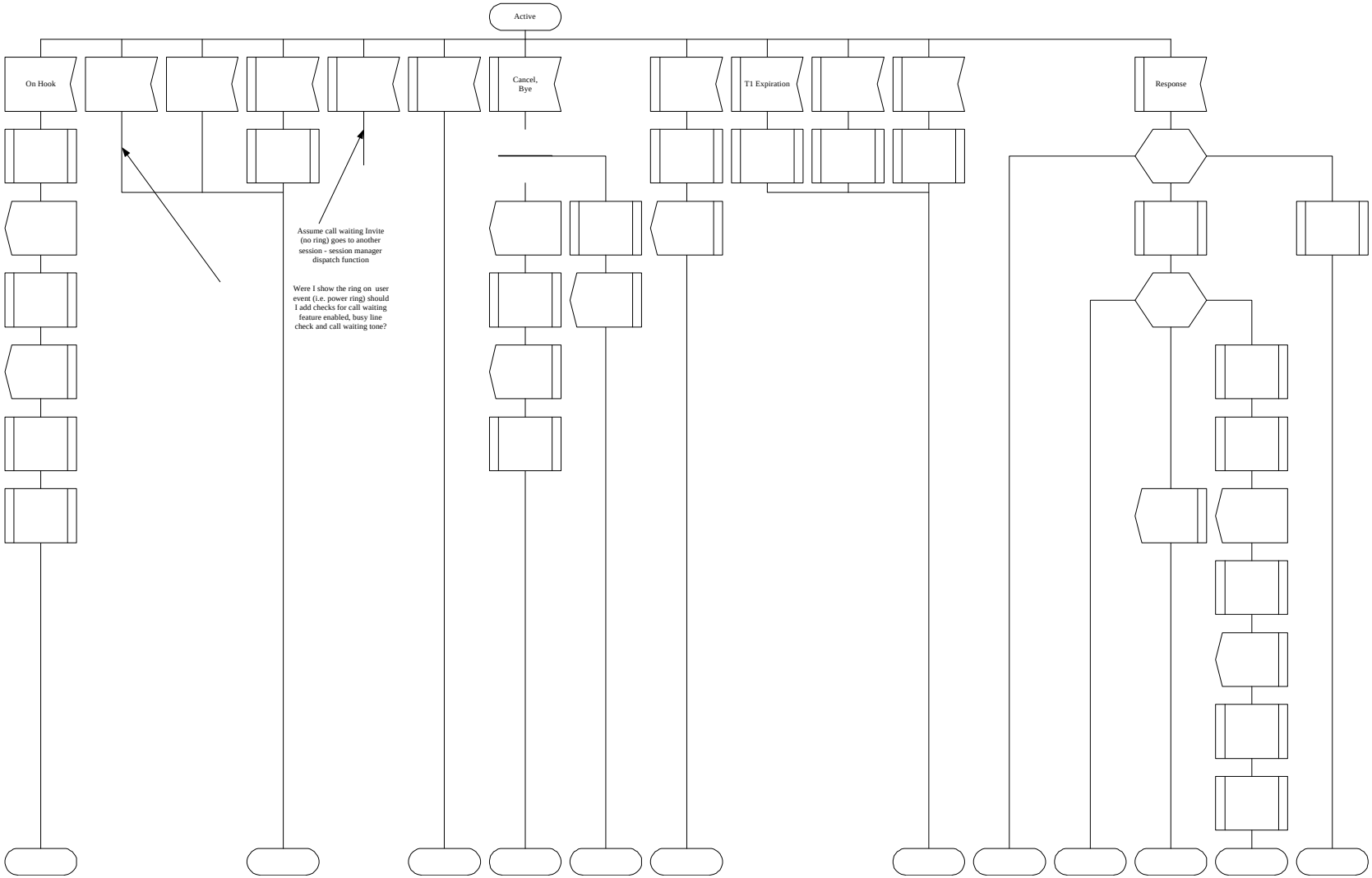


Figure 22: MTA Transitions from Active State

9.17 MTA Transitions from Teardown State

Text to be provided

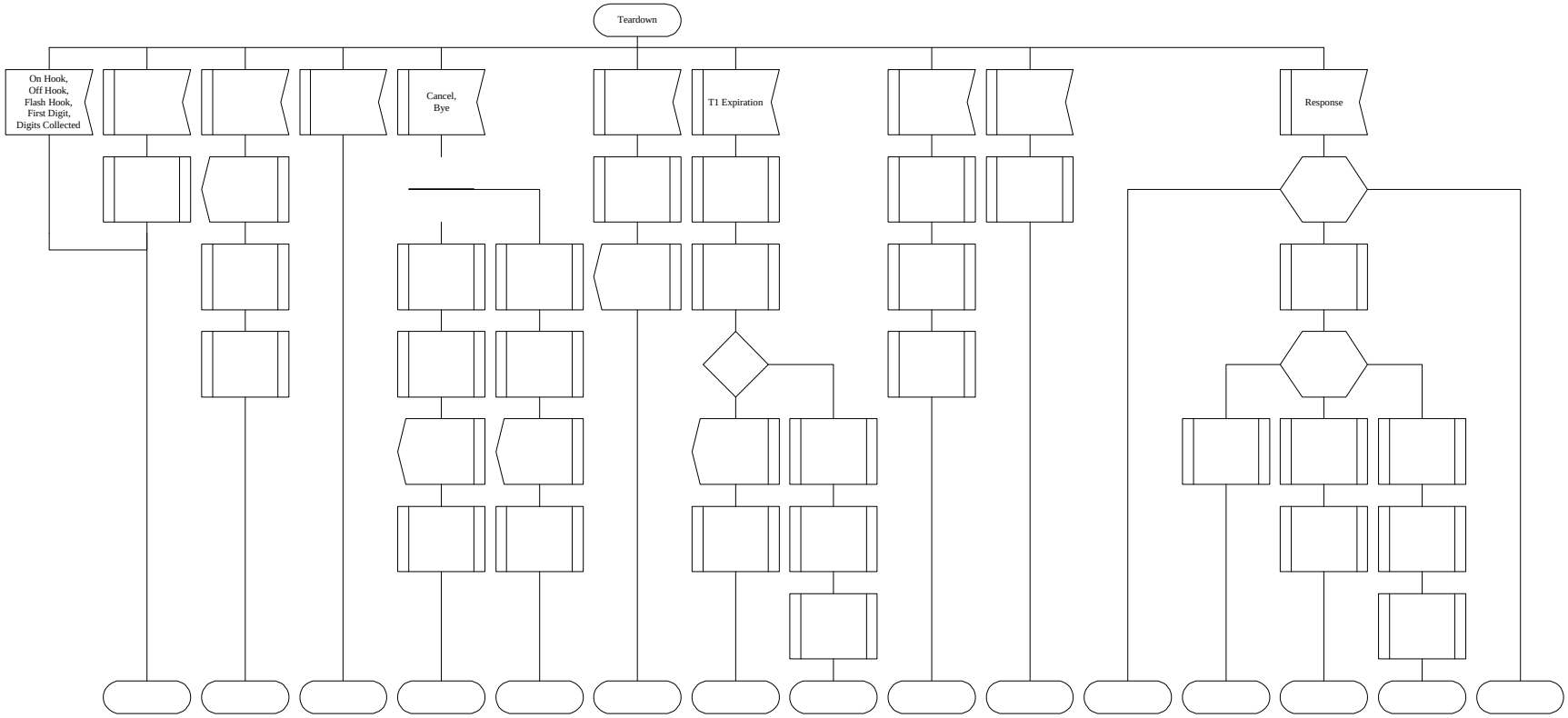


Figure 23: MTA Transitions from Teardown State

9.18 MTA Transitions from Cancel State

Text to be provided

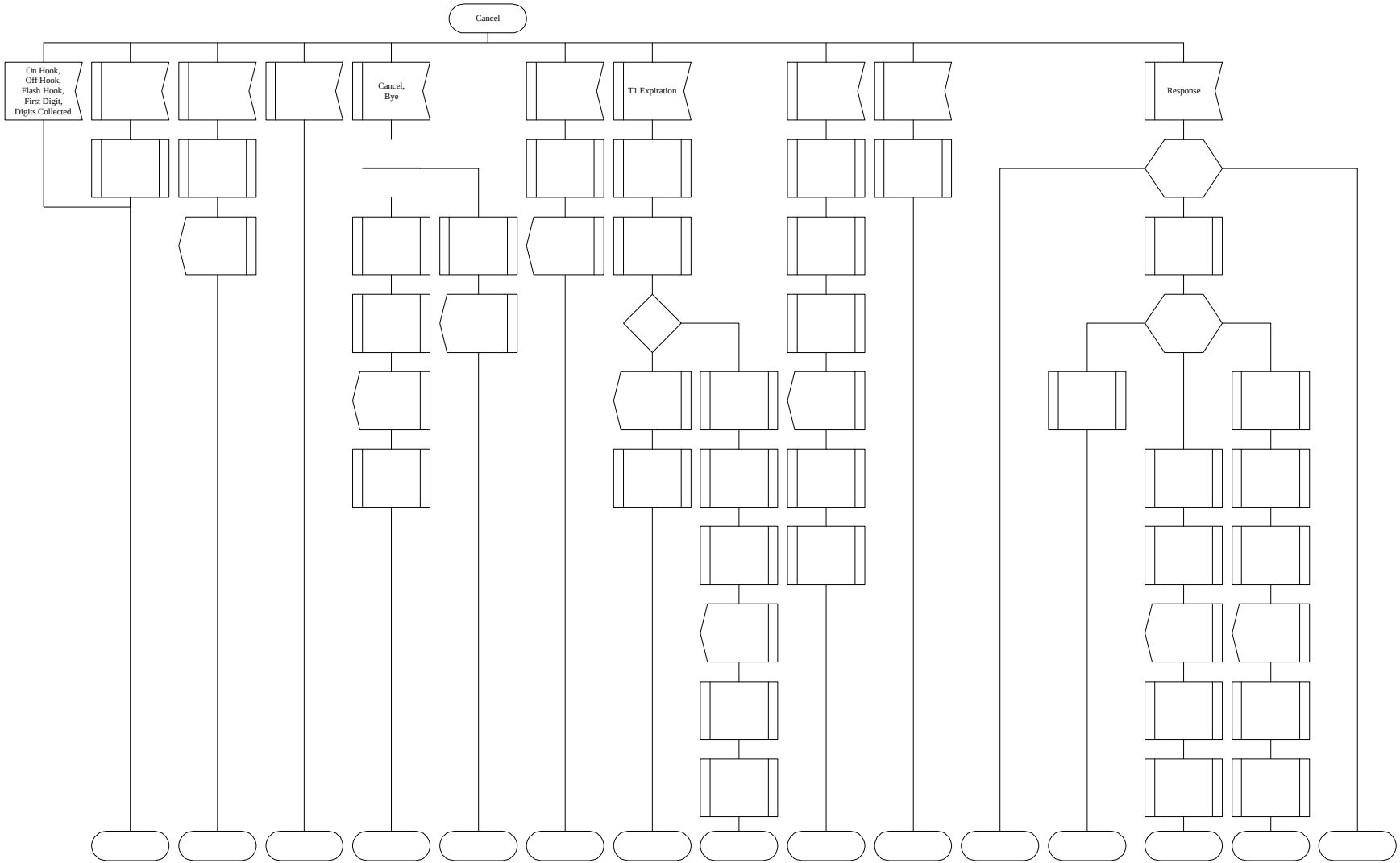


Figure 24: MTA Transitions from Cancel State

9.19 MTA Transitions from Failure State

Text to be provided

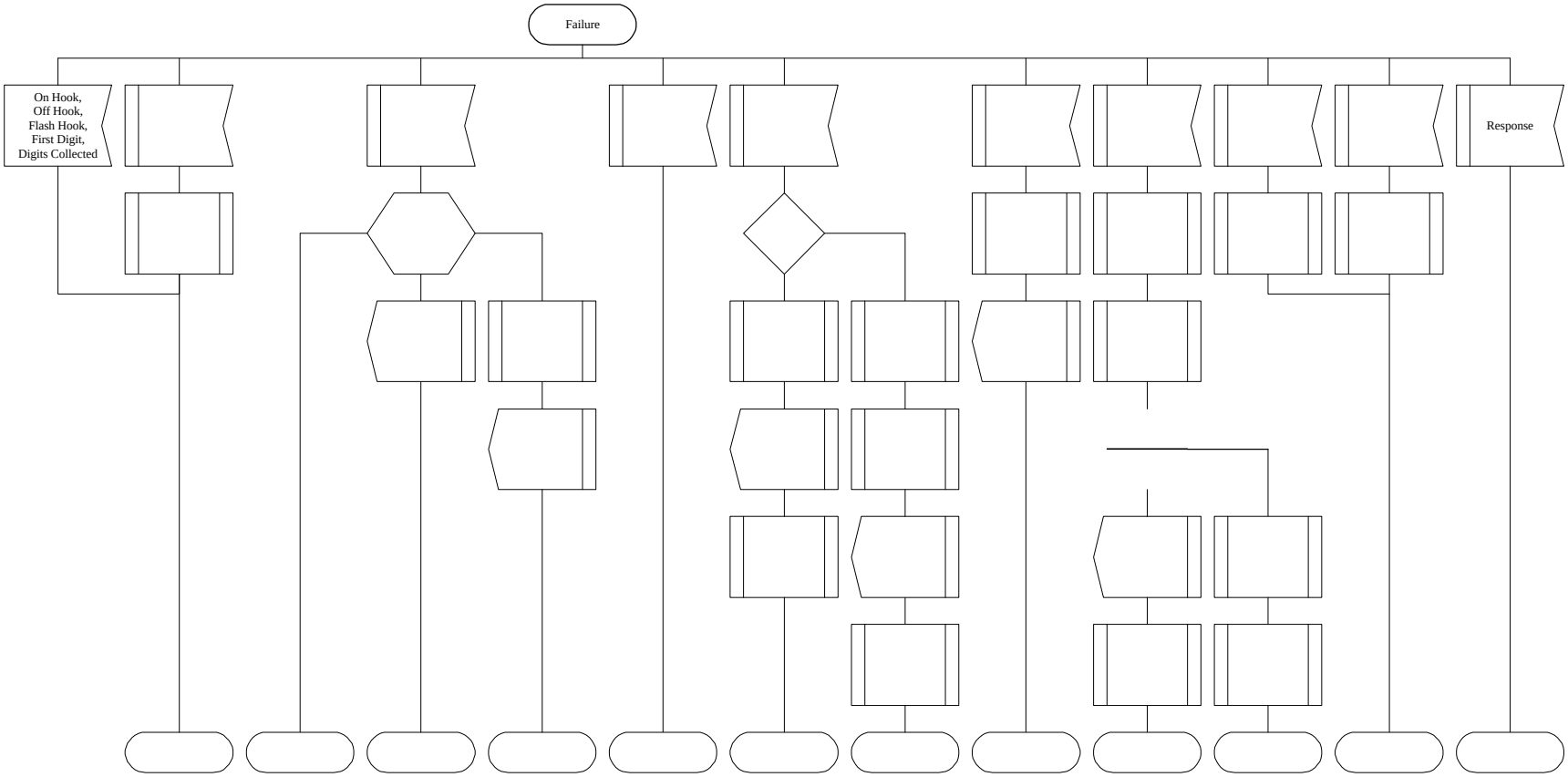


Figure 25: MTA Transitions from Failure State

9.20 MTA Transitions from Wait For On-Hook State

Text to be provided

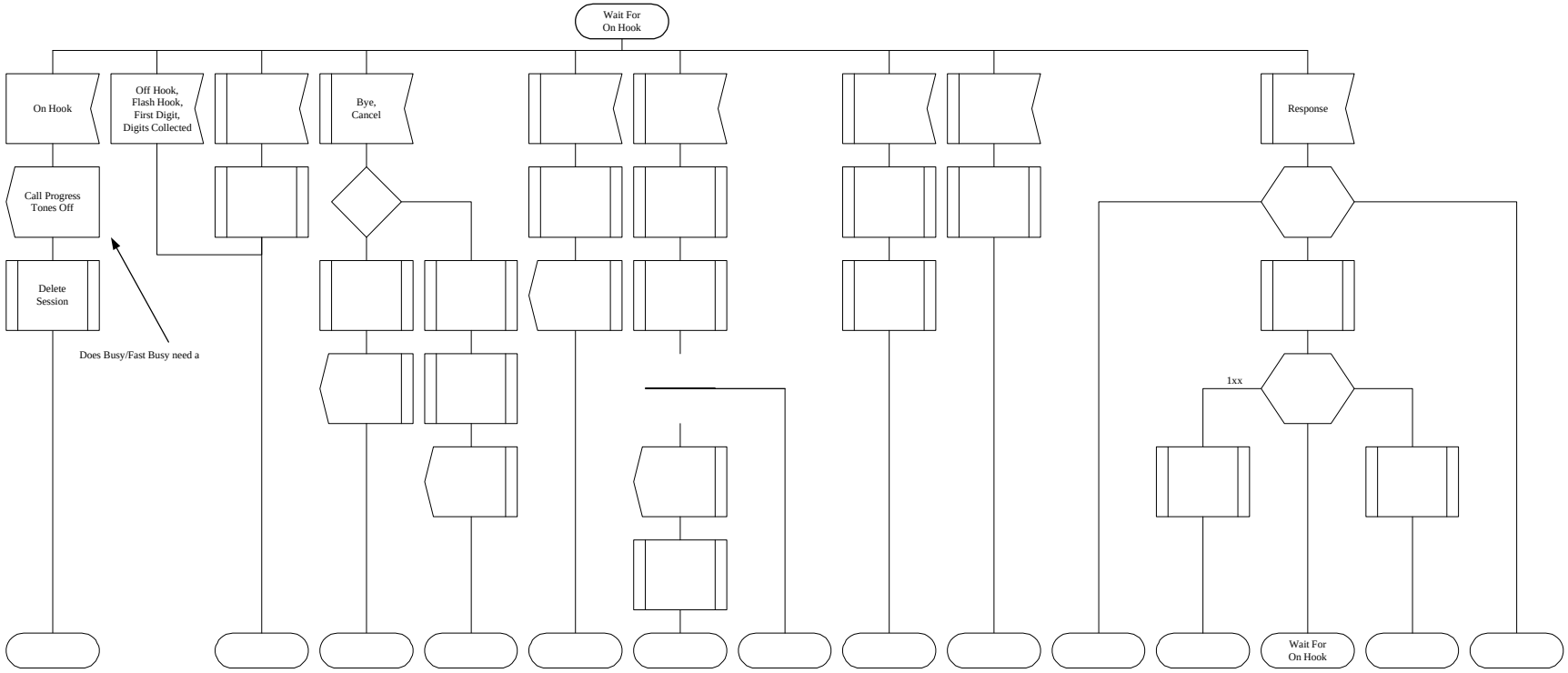


Figure 26: MTA Transitions from Wait For On-Hook State

9.21 MTA Transitions from Call-Forwarding State

Text to be provided

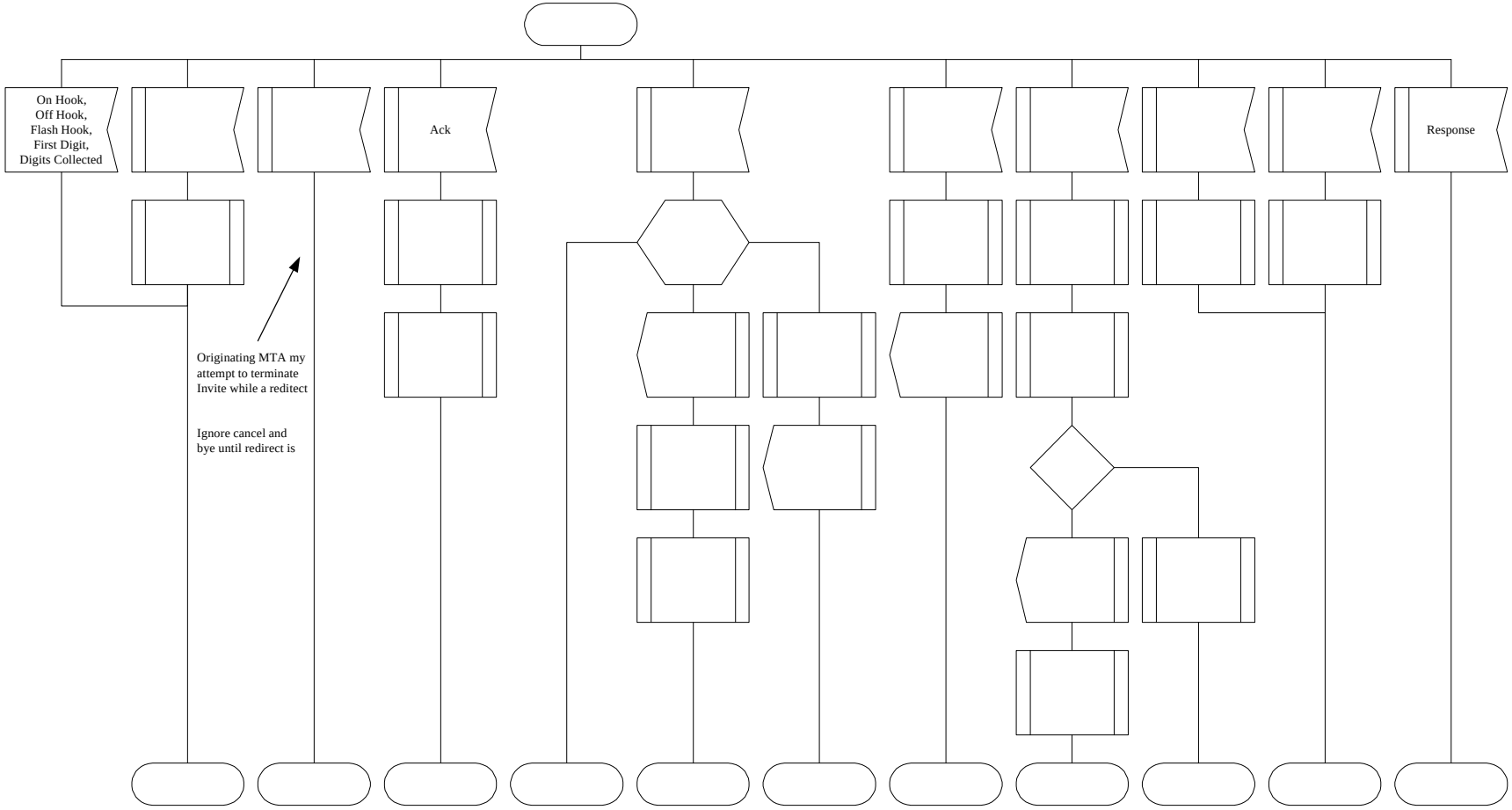


Figure 27: MTA Transitions from Call-Forwarding State

9.22 MTA Transitions from Register State

Text to be provided

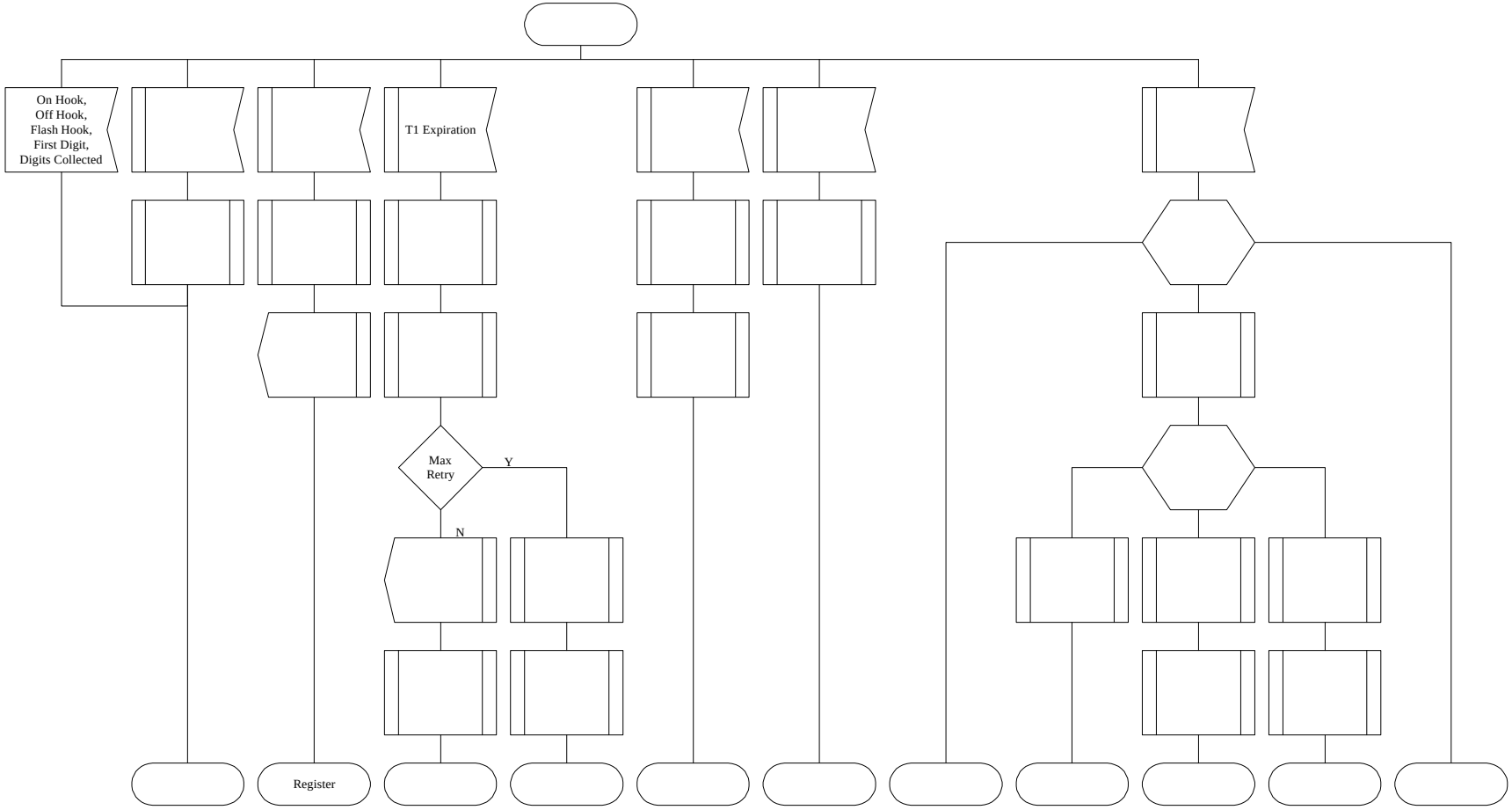


Figure 28: MTA Transitions from Register State

10. SDL Description of CMS/Proxy

This section is still work in progress, and is not considered normative at this time.

TBD

11. SDL Description of CMS/Agent

This section is still work in progress, and is not considered normative at this time.

TBD

Appendix A Timer Summary

This appendix summarizes the timers in the Call Signaling Specification. All timer durations should be network settable based upon the software downloaded into the MTA. The durations given are only meant to illustrate the order of magnitude duration of the timer.

Timer Label	Approximate Duration	Timer Description
Initial Retransmission timers at MTAs, CMS/Agents, and at CMS/Proxies		
T-proxy-request	500 ms	Timer between a MTA sending a request to a CMS/Proxy (e.g., an INVITE), and receiving a valid response, either provisional or final. If T-proxy-request expires before receiving a response, the MTA resends the request. Timer between a CMS/Proxy sending a request to another CMS or to an MTA (e.g. an INVITE) and receiving a valid response, either provisional or final. If T-proxy-request expires before receiving a response, the CMS/Proxy resends the request. Timer between a CMS/Agent sending a request to another CMS (e.g. an INVITE) and receiving a valid response, either provisional or final. If T-proxy-request expires before receiving a response, the CMS/Agent resends the request.
T-proxy-response	500 ms	Timer between a MTA sending a response to a CMS/Proxy (e.g. 200-OK to an INVITE, or a provisional response requesting an acknowledgement), and receiving a valid acknowledgement (e.g. ACK or PRACK). If T-proxy-response expires before receiving a response, the MTA resends the response. Timer between a CMS/Agent sending a response to another CMS (e.g. 200-OK to an INVITE, or a provisional response requesting an acknowledgement), and receiving a valid acknowledgement (e.g. ACK or PRACK). If T-proxy-response expires before receiving a response, the CMS/Agent resends the response.
T-direct-request	500 ms	Timer between a MTA or CMS/Agent sending a request directly to another MTA or CMS/Agent (e.g. an INVITE), and receiving a valid response. If T-direct-request expires before receiving a response, the MTA or CMS/Agent resends the request.
T-direct-response	500 ms	Timer between a MTA or CMS/Agent sending a response directly to another MTA or CMS/Agent (e.g., a 200-OK message), and receiving a valid

		acknowledgement. If T-direct-response expires before receiving a response, the MTA or CMS/Agent resends the response.
Session timers (T3) at originating MTAs and CMS/Agents		
T-setup	30 seconds	Timer between receiving a provisional response to an INVITE and receiving a 180/3-Ringing or 200-OK final response. If T-setup expires before receiving the ring or final response, the MTA or CMS/Agent sends a CANCEL and aborts the call attempt.
T-ringback	5 to 6 minutes	Timer between beginning ringback and receiving a 200-OK indicating the call has been answered. If T-ringback expires before connect, the MTA or CMS/Agent sends a CANCEL message and releases the reserved resources. T-ringback should be sufficiently larger than T-ringing to allow for clock skew.
Session timers (T3) at terminating MTAs and CMS/Agents		
T-resource	10 seconds	Timer between receiving an INVITE request and receiving a PRECONDITION-MET message. If T-resource expires before receiving the PRECONDITION-MET, the MTA or CMS/Agent aborts the incoming call attempt.
T-ringing	3 to 4 minutes	Timer between beginning to ring the phone and connect. If T-ringing expires before connect, the MTA or CMS/Agent sends a 480-Temporarily-Unavailable response and releases the reserved resources, or invoke features such as call forwarding no-answer.
Session timers (T3) at CMS/Proxy		
T-proxy-setup	30 seconds	Timer between sending an INVITE request and receiving a 183-Session-Progress response. This is the maximum length of time the proxy needs to store the call state in DCS; on receipt of the 183-Session-Progress the proxy can revert to a SIP stateless proxy. If T-proxy-setup expires before receiving the 183-Session-Progress, the CMS/Proxy sends a CANCEL and aborts the call attempt.

Table 11-1: Timer Summary

Appendix B Basic Call Flow - MTA to MTA

Figure 29 shows the basic DCS call flow from one MTA to another. The basic DCS call flow starts with an INVITE from MTA_O to MTA_T through CMS/Proxies ($CMS/Proxy_O$ and $CMS/Proxy_T$). It follows the conventions of SIP. The Via headers are used to track the path of the request (INVITE) so that the response can traverse backwards through the same path.

This INVITE is sent requesting that MTA_T should not ring until the QoS preconditions are met. The purpose of this first INVITE is to invoke call features, such as call forwarding, to determine the proper destination MTA, and to negotiate the bandwidth and codec to be used so that the proper resources can be reserved. The response (183-Session-Progress) acknowledges the receipt of the INVITE message, provides the SDP for the forward media flow, and provides contact information for end-to-end messages that happen later in the call flow. When the INVITE is received, MTA_T's state reflects that a call is now being set-up. After MTA_O receives the 183-Session-Progress, it sends a PRACK message directly to MTA_T (as specified in the contact header) to acknowledge receipt of MTA_T's SDP.

After resources are reserved for the call, a PRECONDITION-MET is sent to MTA_T. MTA_T responds with a 200-OK, and also sends a ringback indication in the form of a 180-RINGING message. When the call is answered, a 200-OK to the INVITE is sent back to MTA_O, which ACKs the OK to MTA_T to complete the triple handshake.

The bearer channel session can now be established. When the call is over, either end can send a BYE message directly to the other end. This BYE request must also be responded to with a 200-OK.

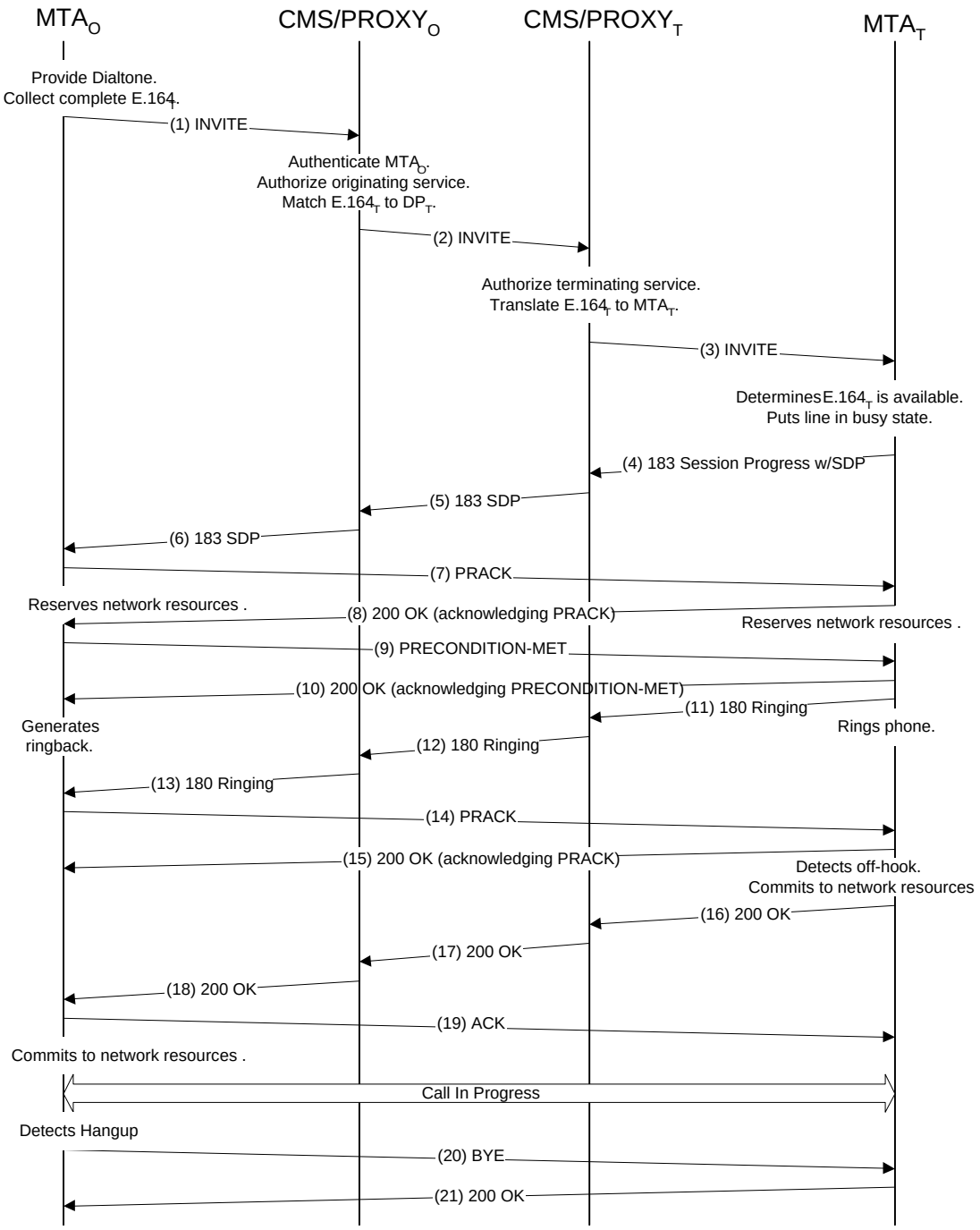


Figure 29: MTA to MTA Call Signaling Flow

A call setup begins when MTA_O detects off-hook on one of its lines. MTA_O first puts that line in the "busy" state. MTA_O sends an audible dialtone signal to the customer and begins to detect DTMF digits. Upon receiving the first digit, MTA_O stops dialtone. Once a complete E.164 number has been received (based upon a digit map that has been provisioned in the MTA), MTA_O generates the following SIP INVITE message and sends it to $CMS/Proxy_O$ (the CMS/Proxy that manages MTA_O). MTA_O starts the retransmission timer (T-proxy-request).

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(DP-o);user=phone SIP/2.0	*Request URI starts with the dialed number from the user*
Via: SIP/2.0/UDP Host(mta-o.provider)	*IP Address or Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	*Calling name and number, as provided by MTA*
Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) uniquely identifies the call-leg, excluding the display-name in the From: header.. To maintain privacy, the addr-spec is encrypted and calling-number and calling-name will be omitted from MTA-MTA signaling.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*To: is a cryptographical hash of a string that contains the dialed digits from the user, timestamp, and a sequence number, or other random string.*
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	*Call-ID is a cryptographically random identifier.*
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-o.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuite:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc- codecs:96	

Upon receiving the INVITE message, CMS/Proxy$_O$ authenticates MTA$_O$ using standard IPSec authentication. CMS/Proxy$_O$ examines the Dcs-Remote-Party-ID: line and checks to see that this originating phone number belongs to MTA$_O$, and is authorized for originating service. CMS/Proxy$_O$ also checks to make sure the calling name in the Dcs-Remote-Party-ID: line is a valid calling name for this line. CMS/Proxy$_O$ then sends the dialed number to a directory server for resolution to an IP address. In this example, the directory server returns the address of CMS/Proxy$_T$, the CMS/Proxy that manages the terminating MTA. CMS/Proxy$_O$ generates the following INVITE message and sends it to CMS/Proxy$_T$. CMS/Proxy$_O$ adds a number of parameters to the INVITE message, which are described below. Upon sending this INVITE message, CMS/Proxy$_O$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(2) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(dp-t);user=np-query SIP/2.0	*"lrn" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(DP-o.provider);branch=1	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis required for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing*

Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS/Proxy-o*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE message, CMS/Proxy$_T$ authenticates that the sender was CMS/Proxy$_O$ using IPSec, and sends the E.164$_T$ address to the directory server. In this example, the Directory Server is able to translate E.164$_T$ to the IP address of MTA$_T$ (one of the MTAs managed by CMS/Proxy$_T$). CMS/Proxy$_T$ then checks to see if MTA$_T$ is authorized for receiving this call. CMS/Proxy$_T$ also checks the account information to determine if MTA$_O$ is paying for the call or if MTA$_T$ is expected to pay. CMS/Proxy$_T$ generates the following INVITE message and sends it to MTA$_T$. The Dcs-Remote-Party-ID line appears unchanged only if the destination MTA has subscribed to caller-id service; otherwise, or if the caller had specified privacy of the caller information, the Dcs-Remote-Party-ID line would be altered. Note that the Via lines have been encrypted, maintaining the privacy of the caller. The line Dcs-State has been added, and contains all the information needed by the CMS/Proxy for any subsequent call features that may be requested. This information is signed by CMS/Proxy$_T$ and encrypted.

Upon sending this INVITE message, CMS/Proxy$_T$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(3) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA$_T$.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*
Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	

Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, MTA_T authenticates that the message came from $CMS/Proxy_T$ using IPSec. MTA_T checks the telephone line associated with the $E.164_T$ (as found in the Request URI) to see if it is available. If it is available, MTA_T looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA_T stores the INVITE message, including the encrypted Dcs-State parameters, for later use. MTA_T puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to $CMS/Proxy_T$. MTA_T starts the retransmission timer with value (T-proxy-response) and starts the session timer (T3) with value (T-resource).

MTA_T can, at its option, still accept further incoming calls and present them all to the customer. Such enhanced user interfaces for the MTA is beyond the scope of this specification. Note that MTA_T can't use the To: header field to determine the proper line, as it may be totally unrelated to the phone number at MTA_T.

(4) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as presented in INVITE message.*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored in MTA_T for this session.*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Called name and number, as provided by MTA*
Dcs-Anonymity: off	*Called name and number privacy is not requested for this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	*Request for acknowledgement of this provisional response*
Session: qos	
Contact: sip:Host(mta-t.provider)	*Address for future direct signaling messages to MTA_T*
Content-Type: application/sdp	*The response to INVITE in SIP must contain the SDP description of the media stream to be sent to MTA_T.*
Content-length: (…)	
v=0	*SDP contains the MTA_T bearer channel IP address, and negotiated voice encoding parameters*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	

a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_T$ forwards the following message to CMS/Proxy$_O$, restoring the Via headers, and adding Dcs-Gate information. At this point CMS/Proxy$_T$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy. CMS/Proxy$_T$ may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(5) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	*State information for CMS/Proxy$_O$ included in the INVITE message*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	*IP address of the terminating gate (CMTS$_T$ IP address), Gate-ID, and security key to enable gate-coordination in Dynamic QoS*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	*Authenticated id of called party*
Dcs-Anonymity: off	
From: “Alien Blaster” <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_O$ forwards the following message to MTA$_O$. This message contains a Dcs-State parameter giving all the information needed by the CMS/Proxy for later features. The Dcs-State value is signed by CMS/Proxy$_O$ and encrypted by CMS/Proxy$_O$'s privately-held key. At this point CMS/Proxy$_O$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy.

(6) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	
Dcs-Media-Authorization: 17S30124	*ID of gate at originator end of connection*

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_O$ private key containing: E.164$_O$; E.164$_T$; CMTS$_O$ IP address:port and Gate-ID, and routing to destination MTA*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a-X=pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, MTA$_O$ stops timer (T-proxy-request) and sends the following PRACK message directly to MTA$_T$ using the IP address in the Contact header of the 183-Session-Progress message.

(7) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a-X-pc-qos:mandatory sendrecv	

MTA$_T$ acknowledges the PRACK with a 200-OK, and begins to reserve the resources necessary for the call.

(8) 200 OK:	***Description***

SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

After sending PRACK(7), MTA_O attempts to reserve network resources if necessary. If resource reservation is successful, MTA_O sends the following PRECONDITION-MET message directly to MTA_T. MTA_O starts timer (T-direct-request).

(9) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 200-OK message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (...)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

MTA_T acknowledges the PRECONDITION-MET message with a 200-OK.

(10) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(10), MTA_O stops timer (T-direct-request).

Upon receipt of the (7) PRACK message, MTA_T stops timer (T-proxy-response) and attempts to reserve network resources if necessary. Once MTA_T both receives the PRECONDITION-MET message and has successfully reserved network resources, MTA_T begins to send ringing voltage to the designated line and sends the following 180 RINGING message through CMS/$Proxy_T$. MTA_T restarts the session timer (T3) with value (T-ringing).

(11) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	

Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored in MTA$_T$ for this session.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

CMS/Proxy$_T$ decodes the Via: headers, and passes the 180-Ringing to CMS/Proxy$_O$. This operation is done as a SIP stateless proxy.

(12) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

CMS/Proxy$_O$ handles the message as a SIP stateless proxy, and passes the 180-Ringing to MTA$_O$.

(13) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 180 RINGING message, MTA$_O$ restarts the transaction timer (T3) with value (T-ringing). MTA$_O$ acknowledges the provisional response with a PRACK, and plays audible ringback tone to the customer.

(14) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 130 PRACK	*New Cseq value for this message*
Rseq: 9022 127 INVITE	*Message being acknowledged*

MTA$_T$ acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(15) 200 OK:	***Description***

SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once MTA_T detects off-hook on the called line, it disconnects ringing voltage from the line and sends the final response through the proxies. MTA_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, MTA_T may also commit to resources that have been reserved for this call. At this point, MTA_T begins to generate bearer channel packets of encoded voice and send them to MTA_O using the IP address and port number specified in the SDP part of the original INVITE message.

(16) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored in MTA_T for this session.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Proxy_T$ handles the message as a SIP stateless proxy, and forwards it to $CMS/Proxy_O$.

(17) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Proxy_O$ handles the message as a SIP stateless proxy, and forwards it to MTA_O.

(18) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, MTA_O stops timer (T-ringing) and stops playing audible ringback tone to the customer and begins to play the bearer channel stream that is received from MTA_T. MTA_O sends the following ACK message to MTA_T. If necessary, MTA_O may also commit to resources that have been reserved for this call. At this point, MTA_O begins to generate bearer channel packets of encoded

voice and send them to MTA_T using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(19) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, MTA_T stop timer (T-proxy-response).

When either MTA detects hangup, it sends out a BYE message to the other MTA. In this example, MTA_O detected that the customer hung up the phone. MTA_O puts that line in the "idle" state so new calls can be made or received. It sends the following BYE message directly to MTA_T. MTA_O may also need to release network resources that have been used for the call. MTA_O starts timer (T-direct-request).

(20) BYE:	***Description***
BYE sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of the BYE message, MTA_T stops playing the bearer channel stream received from MTA_O and, if necessary, releases network resources that have been used for this call. MTA_T sends the following 200-OK message to MTA_O. MTA_T starts a 15-second timer (T-hangup) (Note: this is a local interface issue, and not part of this specification). If MTA_T does not detect hangup on the line before timer (T-hangup) expires, it plays "reorder" tone on the customer line. Once hangup is detected, MTA_T puts that line in the "idle" state so new calls can be made or received.

(21) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of 200-OK, MTA_O stops timer (T-direct-request).

Appendix C Basic Call Flow from MTA to CMS /Agent

This section describes the DCS call signaling flow for a basic call that terminate on the PSTN, or some other endpoint controlled by a CMS/Agent.

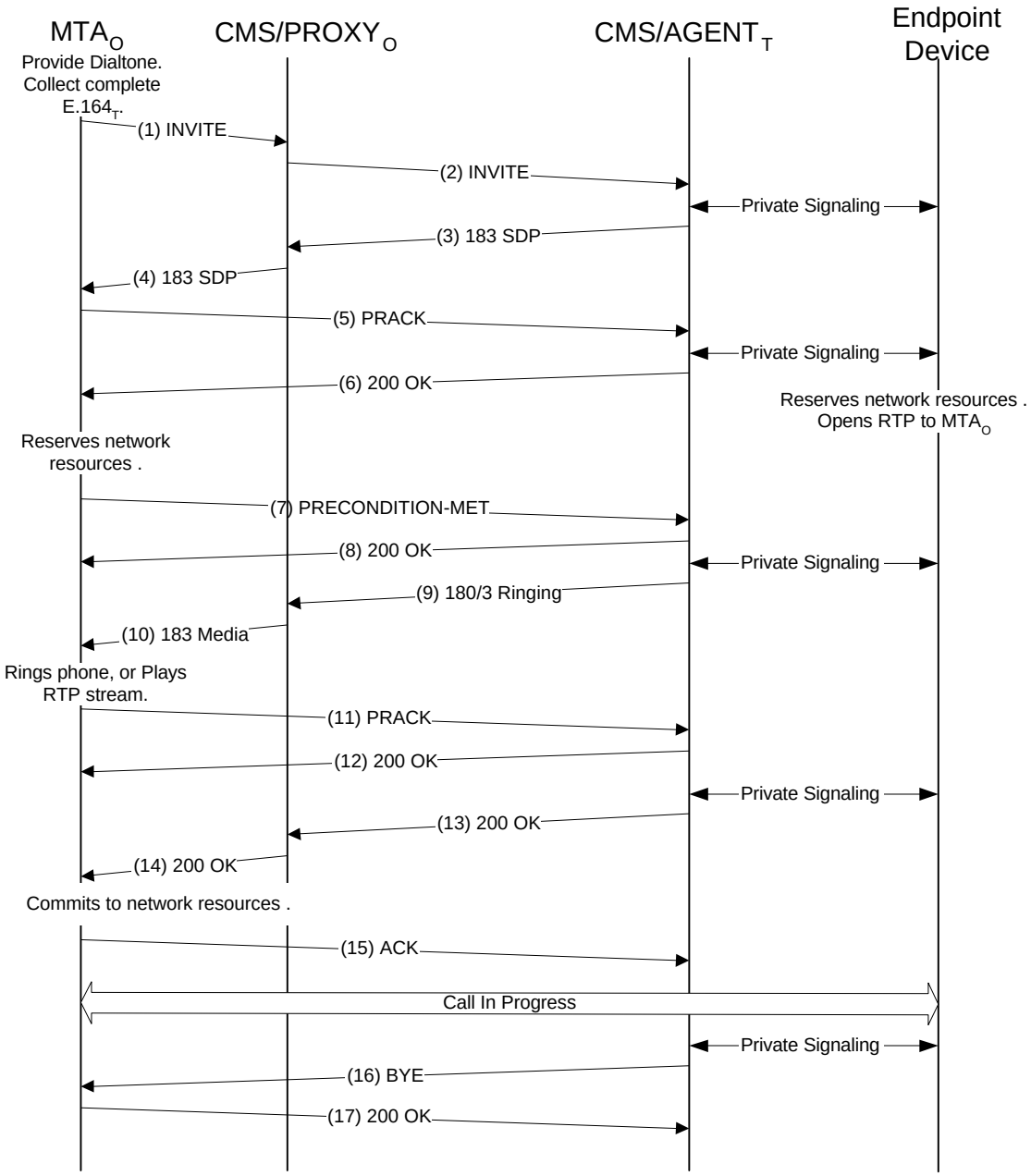


Figure 30: MTA to CMS/Agent Call Flow

A call setup begins when MTA$_O$ detects off-hook on one of its lines. MTA$_O$ first puts that line in the "busy" state. MTA$_O$ sends an audible dialtone signal to the customer and begins to detect DTMF digits.

Upon receiving the first digit, MTA_O stops dialtone. Once a complete E.164 number has been received (based upon a digit map that has been provisioned in the MTA), MTA_O generates the following SIP INVITE message and sends it to $CMS/Proxy_O$ (the CMS/Proxy that manages MTA_O). MTA_O starts the retransmission timer (T-proxy-request).

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(DP-o);user=phone SIP/2.0	*Request URI starts with the dialed number from the user*
Via: SIP/2.0/UDP Host(mta-o.provider)	*IP Address or Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	*Calling name and number, as provided by MTA*
Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) uniquely identifies the call-leg, excluding the display-name in the From: header.. To maintain privacy, the addr-spec is encrypted and calling-number and calling-name will be omitted from MTA-MTA signaling.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*To: is a cryptographical hash of a string that contains the dialed digits from the user, timestamp, and a sequence number, or other random string.*
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	*Call-ID is a cryptographically random identifier.*
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-o.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuite:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc- codecs:96	

Upon receiving the INVITE message, $CMS/Proxy_O$ authenticates MTA_O using standard IPSec authentication. $CMS/Proxy_O$ examines the Dcs-Remote-Party-ID: line and checks to see that this originating phone number belongs to MTA_O, and is authorized for originating service. $CMS/Proxy_O$ also checks to make sure the calling name in the Dcs-Remote-Party-ID: line is a valid calling name for this line. $CMS/Proxy_O$ then sends the dialed number to a directory server for resolution to an IP address. In this example, the directory server returns the address of CMS_T, the CMS that manages the endpoint device. $CMS/Proxy_O$ generates the following INVITE message and sends it to CMS_T. $CMS/Proxy_O$ adds a number of parameters to the INVITE message, which are described below. Upon sending this INVITE message, $CMS/Proxy_O$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(2) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(cms-t);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(DP-o.provider);branch=1	*$CMS/Proxy_O$ IP address; branch indicates this is the first destination attempt*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	

Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis required for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS/Proxy-o*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE message, CMS$_T$ authenticates that the sender was CMS/Proxy$_O$ using IPSec, and determines the proper endpoint device to receive this call. CMS$_T$ engages in local signaling with that endpoint device, outside the scope of this specification, and determines the proper SDP for the media flow to this endpoint device. When complete, CMS$_T$ forwards the following message message to CMS/Proxy$_O$. The CMS$_T$ lists itself as the location of the Dcs-Gate, since it simulates the gate operation. CMS$_T$ may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(3) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Gate: Host(cms-t.provider):4321/137S90721/805628	*CMS$_T$ simulates the gate in the CMTS*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
Dcs-Anonymity: off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(cms-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	

s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_O$ forwards the following message to MTA$_O$. This message contains a Dcs-State parameter giving all the information needed by the CMS/Proxy for later features. The Dcs-State value is signed by CMS/Proxy$_O$ and encrypted by CMS/Proxy$_O$'s privately-held key. At this point CMS/Proxy$_O$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy.

(4) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	
Dcs-Media-Authorization: 17S30124	*ID of gate at originator end of connection*
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(cms-t)}$_K$"	*State blob encrypted with a CMS/Proxy$_O$ private key containing: E.164$_O$; E.164$_T$; CMTS$_O$ IP address:port and Gate-ID, and routing to destination MTA*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(cms-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a-X=pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, MTA$_O$ stops timer (T-proxy-request) and sends the following PRACK message directly to CMS$_T$ using the IP address in the Contact header of the 183-Session-Progress message.

(5) PRACK:	***Description***
PRACK sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	

Content-length: (…)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a-X-pc-qos:mandatory sendrecv	

CMS_T acknowledges the PRACK with a 200-OK, and performs local signaling with the endpoint (outside the scope of this specification) in order to begin reserving the resources necessary for the call.

(6) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

After sending PRACK(5), MTA_O attempts to reserve network resources if necessary. If resource reservation is successful, MTA_O sends the following PRECONDITION-MET message directly to CMS_T. MTA_O starts timer (T-direct-request).

(7) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

CMS_T acknowledges the PRECONDITION-MET message with a 200-OK.

(8) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	

From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(8), MTA_O stops timer (T-direct-request).

Upon receipt of the (5) PRACK message, CMS_T stops timer (T-proxy-response) and signals the endpoint device to attempt to reserve the network resources necessary. Once CMS_T both receives the PRECONDITION-MET message and acknowledgement from the endpoint device, CMS_T sends the following 180-Ringing (or 183-Sesison-Progress, with a Session:Media header) message. CMS_T restarts the session timer(T3) with value (T-ringing).

(9) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(cms-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

$CMS/Proxy_O$ handles the message as a SIP stateless proxy, and passes the 180-Ringing (or 183-Session-Progress) to MTA_O.

(10) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(cms-t.provider)	
Cseq: 127 INVITE	
RSeq: 9022	

Upon receipt of the 180 RINGING message, MTA_O restarts the transaction timer (T3) with value (T-ringing). MTA_O acknowledges the provisional response with a PRACK, and plays audible ringback tone to the customer.

(11) PRACK:	***Description***
PRACK sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
RAck: 9022 127 INVITE	*Message being acknowledged*

CMS_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(12) 200 OK:	*Description*
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once CMS_T, via private signaling with the endpoint device, detects off-hook on the called line, it sends the final response to the INVITE. CMS_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, MTA_T may also commit to resources that have been reserved for this call. At this point, the endpoint device begins to generate bearer channel packets of encoded voice and send them to MTA_O using the IP address and port number specified in the SDP part of the original INVITE message.

(13) 200-OK:	*Description*
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Proxy_O$ handles the message as a SIP stateless proxy, and forwards it to MTA_O.

(14) 200-OK:	*Description*
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, MTA_O stops timer (T-ringing) and stops playing audible ringback tone to the customer and begins to play the bearer channel stream that is received. MTA_O sends the following ACK message to CMS_T. If necessary, MTA_O may also commit to resources that have been reserved for this call. At this point, MTA_O begins to generate bearer channel packets of encoded voice and send them to the remote endpoint using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(15) ACK:	*Description*
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, CMS_T stop timer (T-proxy-response).

When MTA_O detects a hangup, or the endpoint device controlled by CMS_T detects a hangup, it sends out a BYE message to the other endpoint. In this example, CMS_T detected that the customer hung up the phone. CMS_T puts that line in the "idle" state so new calls can be made or received. It sends the following BYE message directly to MTA_O. CMS_T may also need to release network resources that have been used for the call. CMS_T starts timer (T-direct-request).

(16) BYE:	***Description***
BYE sip:Host(mta-o.provider) SIP/2.0	*Address from Contact: header of INVITE message*
Via: SIP/2.0/UDP Host(cms-t.provider)	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of the BYE message, MTA_O stops playing the bearer channel stream received from the endpoint device, and, if necessary, releases network resources that have been used for this call. MTA_O sends the following 200-OK message to CMS_T. MTA_O starts a 15-second timer (T-hangup) (Note: this is a local interface issue, and not part of this specification). If MTA_O does not detect hangup on the line before timer (T-hangup) expires, it plays "reorder" tone on the customer line. Once hangup is detected, MTA_O puts that line in the "idle" state so new calls can be made or received.

(17) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-t.provider)	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of 200-OK, CMS_T stops timer (T-direct-request).

Appendix D Basic Call Flow from CMS/Agent to MTA

This example shows a call originating on the PSTN and directed to a destination on the PacketCable network. We assume the same sequence of user behavior as in the basic call flow of Figure 29, only difference being the location of the originator.

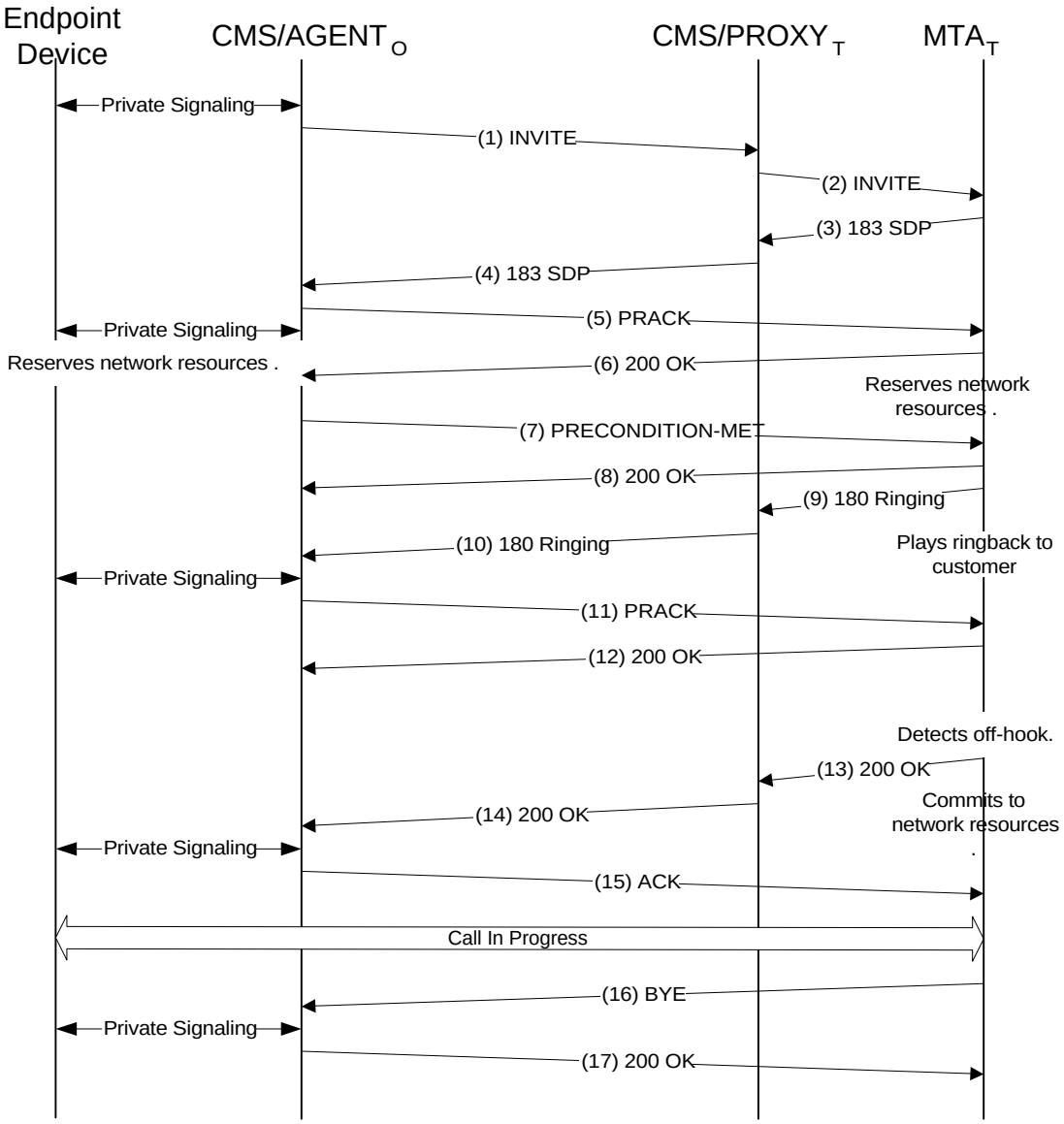


Figure 31: CMS/Agent to MTA Signaling Call Flow

A call setup begins when the endpoint device controlled by CMS_O detects an off-hook condition on one of its lines. This event is communicated to CMS_O through a private signaling exchange beyond the scope of this specification. CMS_O first puts that line in the "busy" state, and collects a complete E.164 number. As a result of a translation function performed by CMS_O, the destination is determined to be a DCS device

served by CMS/Proxy$_T$. CMS$_O$ generates the following SIP INVITE message and sends it to CMS/Proxy$_T$. CMS$_O$ starts the retransmission timer (T-proxy-request) and starts the T3 session timer (T -setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(1) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(DP-t);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(cms-o.provider);branch=1	*CMS$_O$ IP address; branch indicates this is the first destination attempt*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cms-o.provider):3612/17S30124/37FA1948 optional	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis optional for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing*
Dcs-Billing-ID: Host(cms-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: John Doe; <tel:+1-212-555-1111>	*Since anonymity is not being requested, the From and To are not cryptographically random strings, rather endpoint identifying. The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(cms-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS-o*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE message, CMS/Proxy$_T$ authenticates that the sender was CMS$_O$ using IPSec, and sends the E.164$_T$ address to the directory server. In this example, the Directory Server is able to translate E.164$_T$ to the IP address of MTA$_T$ (one of the MTAs managed by CMS/Proxy$_T$). CMS/Proxy$_T$ then checks to see if MTA$_T$ is authorized for receiving this call. CMS/Proxy$_T$ also checks the account information to determine if MTA$_O$ is paying for the call or if MTA$_T$ is expected to pay. CMS/Proxy$_T$ generates the following INVITE message and sends it to MTA$_T$. The Dcs-Remote-Party-ID line appears unchanged only if the destination MTA has subscribed to caller-id service; otherwise, or if the caller had specified privacy of the caller information, the Dcs-Remote-Party-ID line would be altered. Note that the Via lines have been encrypted, maintaining the privacy of the caller. The line Dcs-State has been added, and contains all the information needed by the CMS/Proxy for any subsequent call features that may be requested. This information is signed by CMS/Proxy$_T$ and encrypted.

Upon sending this INVITE message, CMS/Proxy$_T$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(2) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA_T.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); $_K$	*Via headers have been encrypted for originator privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*
Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State blob encrypted with a CMS/$Proxy_T$ privately-held key containing: nexthop routing information, $CMTS_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: John Doe; <tel:+1-212-555-1111>	*Call leg Identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(cms-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, MTA_T authenticates that the message came from CMS/$Proxy_T$ using IPSec. MTA_T checks the telephone line associated with the $E.164_T$ (as found in the Request URI) to see if it is available. If it is available, MTA_T looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA_T stores the INVITE message, including the encrypted Dcs-State parameters, for later use. MTA_T puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to CMS/$Proxy_T$. MTA_T starts the retransmission timer with value (T-proxy-response) and starts the session timer (T3) with value (T-resource).

MTA_T can, at its option, still accept further incoming calls and present them all to the customer. Such enhanced user interfaces for the MTA is beyond the scope of this specification. Note that MTA_T can't use the To: header field to determine the proper line, as it may be totally unrelated to the phone number at MTA_T.

(3) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); $_K$	*Via headers as presented in INVITE message.*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State information stored in MTA_T for this session.*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	
Dcs-Anonymity: off	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	*Request for acknowledgement of this provisional response*
Session: qos	

Contact: sip:Host(mta-t.provider)	*Address for future direct signaling messages to MTA_T*
Content-Type: application/sdp	*The response to INVITE in SIP must contain the SDP description of the media stream to be sent to MTA_T.*
Content-length: (…)	
v=0	*SDP contains the MTA_T bearer channel IP address, and negotiated voice encoding parameters*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, $CMS/Proxy_T$ forwards the following message to CMS_O, restoring the Via headers, and adding Dcs-Gate information. At this point $CMS/Proxy_T$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy. $CMS/Proxy_T$ may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(4) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Dcs-State: Host(dp-t.provider); nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	*State information for $CMS/Proxy_O$ included in the INVITE message*
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	*IP address of the terminating gate ($CMTS_T$ IP address), Gate-ID, and security key to enable gate-coordination in Dynamic QoS*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
Dcs-Anonymity: off	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS_O stops timer (T-proxy-request) and sends the following PRACK message directly to MTA_T using the IP address in the Contact header of the 183-Session-Progress message.

(5) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	

From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a-X-pc-qos:mandatory sendrecv	

MTA_T acknowledges the PRACK with a 200-OK, and begins to reserve the resources necessary for the call.

(6) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

After sending PRACK(5), CMS_O signals to the endpoint device to attempt to reserve the network resources necessary for the connection. If the endpoint signals that resource reservation is successful, CMS_O sends the following PRECONDITION-MET message directly to MTA_T. CMS_O starts timer (T-direct-request).

(7) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

MTA_T acknowledges the PRECONDITION-MET message with a 200-OK.

(8) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(10), CMS_O stops timer (T-direct-request).

Upon receipt of the (5) PRACK message, MTA_T stops timer (T-proxy-response) and attempts to reserve network resources if necessary. Once MTA_T both receives the PRECONDITION-MET message and has successfully reserved network resources, MTA_T begins to send ringing voltage to the designated line and sends the following 180 RINGING message through CMS/$Proxy_T$. MTA_T restarts the session timer(T3) with value (T-ringing).

(9) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); $_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State information stored in MTA_T for this session.*
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

CMS/$Proxy_T$ decodes the Via: headers, and passes the 180-Ringing to CMS_O. This operation is done as a SIP stateless proxy.

(10) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(cms-o.provider);branch=1	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 127 INVITE	
RSeq: 9022	

Upon receipt of the 180 RINGING message, CMS_O restarts the transaction timer (T3) with value (T-ringing). CMS_O acknowledges the provisional response with a PRACK, and signals the endpoint device to play audible ringback tone to the customer.

(11) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 200-OK message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
RAck: 9022 127 INVITE	*Message being acknowledged*

MTA_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(12) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once MTA_T detects off-hook on the called line, it disconnects ringing voltage from the line and sends the final response through the proxies. MTA_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, MTA_T may also commit to resources that have been reserved for this call. At this point, MTA_T begins to generate bearer channel packets of encoded voice and send them to MTA_O using the IP address and port number specified in the SDP part of the original INVITE message.

(13) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); K	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(cms-o.provider); gate=Host(cmts-t.provider):4321/31S14621; via="Host(cms-o.provider);branch=1"}K"	*State information stored in MTA_T for this session.*
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Proxy_T$ handles the message as a SIP stateless proxy, and forwards it to CMS_O.

(14) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, CMS_O stops timer (T-ringing) and signals the endpoint device to stop playing audible ringback tone to the customer and to begin to play the bearer channel stream that is received from MTA_T. CMS_O sends the following ACK message to MTA_T. If necessary, CMS_O may also commit to resources that have been reserved for this call. At this point, the endpoint device begins to generate bearer channel packets of encoded voice and send them to MTA_T using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(15) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, MTA_T stop timer (T-proxy-response).

When either endpoint detects hangup, it sends out a BYE message to the other one. In this example, MTA_T detected that the customer hung up the phone. MTA_T puts that line in the "idle" state so new calls can be

made or received. It sends the following BYE message directly to CMS_O. MTA_T may also need to release network resources that have been used for the call. MTA_T starts timer (T-direct-request).

(16) BYE:	***Description***
BYE sip:Host(cms-o.provider) SIP/2.0	*Address from Contact: header of INVITE message*
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: tel:+1-212-555-2222	*Call leg identification. Note that the From: and To: headers are reversed, since this request is coming from the called party.*
To: John Doe; <tel:+1-212-555-1111>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of the BYE message, CMS_O signals the endpoint device to stop playing the bearer channel stream received from MTA_T and, if necessary, releases network resources that have been used for this call. CMS_O sends the following 200-OK message to MTA_T. Once hangup is detected on the endpoint device, CMS_O puts that line in the "idle" state so new calls can be made or received.

(17) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: tel:+1-212-555-2222	
To: John Doe; <tel:+1-212-555-1111>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of 200-OK, MTA_T stops timer (T-direct-request).

Appendix E Basic Call Flow CMS/Agent to CMS/A g ent

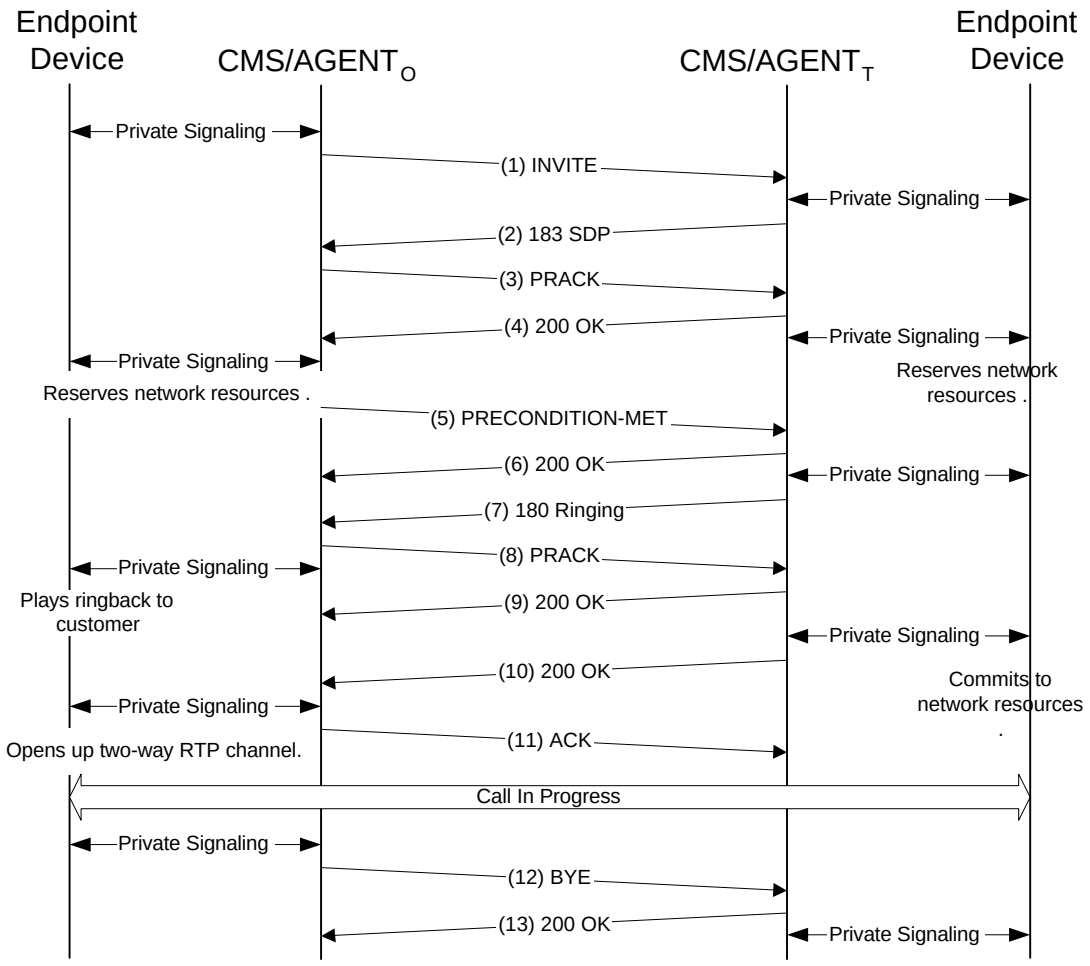


Figure 32: CMS/Agent to CMS/Agent Call Flow

A call setup begins when the endpoint device controlled by CMS_O detects an off-hook condition on one of its lines. This event is communicated to CMS_O through a private signaling exchange beyond the scope of this specification. CMS_O first puts that line in the "busy" state, and collects a complete E.164 number. As a result of a translation function performed by CMS_O, the destination is determined to be an endpoint device served by CMS_T. CMS_O generates the following SIP INVITE message and sends it to CMS_T. CMS_O starts the retransmission timer (T-proxy-request) and starts the T3 session timer (T-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(1) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(cms-t);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(cms-o.provider);branch=1	*CMS_O IP address; branch indicates this is the first destination attempt*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	

Dcs-Gate: Host(cms-o.provider):3612/17S30124/37FA1948 optional	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis optional for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing*
Dcs-Billing-ID: Host(cms-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: John Doe; <tel:+1-212-555-1111>	*Since anonymity is not being requested, the From and To are not cryptographically random strings, rather endpoint identifying.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
Cseq: 127 INVITE	
Contact: sip:Host(cms-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS$_O$*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE message, CMS_T authenticates that the sender was CMS_O using IPSec, and sends the $E.164_T$ address to the directory server. In this example, the Directory Server is able to translate $E.164_T$ to the IP address of one of the endpoint devices controlled by CMS_T. CMS_T then checks to see if the endpoint device is authorized for receiving this call. CMS_T also checks the account information to determine if the originator is paying for the call or if the destination is expected to pay. CMS_T engages in private signaling exchange with the endpoint device, beyond the scope of this specification, and determines the SDP description of the media stream to be sent to this endpoint.

CMS_T puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to CMS_O. The Dcs-Gate header is omitted from this message, since CMS_O indicated it was optional, and CMS_T considers it optional as well. CMS_T starts the retransmission timer with value (T-proxy-response) and starts the session timer (T3) with value (T-resource). CMS_T may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(2) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(cms-o.provider);branch=1	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(cms-t.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(rgw12.provider)	

b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS_O stops timer (T-proxy-request) and sends the following PRACK message to CMS_T using the IP address in the Contact header of the 183-Session-Progress message.

(3) PRACK:	***Description***
PRACK sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a-X-pc-qos:mandatory sendrecv	

CMS_T acknowledges the PRACK with a 200-OK, and signals the endpoint device to begin to reserve the resources necessary for the call.

(4) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

After sending PRACK(3), CMS_O signals to the endpoint device to attempt to reserve the network resources necessary for the connection. If the endpoint signals that resource reservation is successful, CMS_O sends the following PRECONDITION-MET message to CMS_T. CMS_O starts timer (T-direct-request).

(5) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (...)	

v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg02.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

CMS_T acknowledges the PRECONDITION-MET message with a 200-OK.

(6) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(6), CMS_O stops timer (T-direct-request).

Upon receipt of the (3) PRACK message, CMS_T stops timer (T-proxy-response) and attempts to reserve network resources if necessary. Once CMS_T both receives the PRECONDITION-MET message and has successfully reserved network resources, CMS_T signals the endpoint to begin to send ringing voltage to the designated line and sends the following 180 RINGING message. CMS_T restarts the session timer (T3) with value (T-ringing).

(7) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(cms-o.provider);branch=1	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(cms-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

Upon receipt of the 180 RINGING message, CMS_O restarts the transaction timer (T3) with value (T-ringing). CMS_O acknowledges the provisional response with a PRACK, and signals the endpoint device to play audible ringback tone to the customer.

(8) PRACK:	***Description***
PRACK sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
RAck: 9022 127 INVITE	*Message being acknowledged*

CMS_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(9) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once CMS_T detects off-hook on the called line, it disconnects ringing voltage from the line and sends the final response. CMS_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, CMS_T may also commit to resources that have been reserved for this call. At this point, CMS_T signals to the endpoint device to begin to generate bearer channel packets of encoded voice and send them to the originating endpoint, at the IP address and port number specified in the SDP part of the original INVITE message.

(10) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider); branch=1	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, CMS_O stops timer (T-ringing) and signals the endpoint device to stop playing audible ringback tone to the customer and to begin to play the bearer channel stream that is received from the destination endpoint. CMS_O sends the following ACK message to CMS_T. If necessary, CMS_O may also commit to resources that have been reserved for this call. At this point, the endpoint device begins to generate bearer channel packets of encoded voice and send them to the destination endpoint using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(11) ACK:	***Description***
ACK sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, CMS_T stop timer (T-proxy-response).

When either endpoint detects hangup, it sends out a BYE message to the other one. In this example, the originating endpoint detected that the customer hung up the phone. CMS_O puts that line in the "idle" state so new calls can be made or received. It sends the following BYE message directly to CMS_T. CMS_O starts timer (T-direct-request).

(12) BYE:	***Description***
BYE sip:Host(cms-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	*Call leg identification.*
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of the BYE message, CMS_T signals the endpoint device to stop playing the bearer channel stream received from the originator and, if necessary, releases network resources that have been used for this call. CMS_T sends the following 200-OK message to CMS_O. Once hangup is detected on the endpoint device, CMS_T puts that line in the "idle" state so new calls can be made or received.

(13) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(cms-o.provider)	
From: John Doe; <tel:+1-212-555-1111>	
To: tel:+1-212-555-2222	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of 200-OK, CMS_O stops timer (T-direct-request).

Appendix F Call Forwarding Unconditional Call Flow

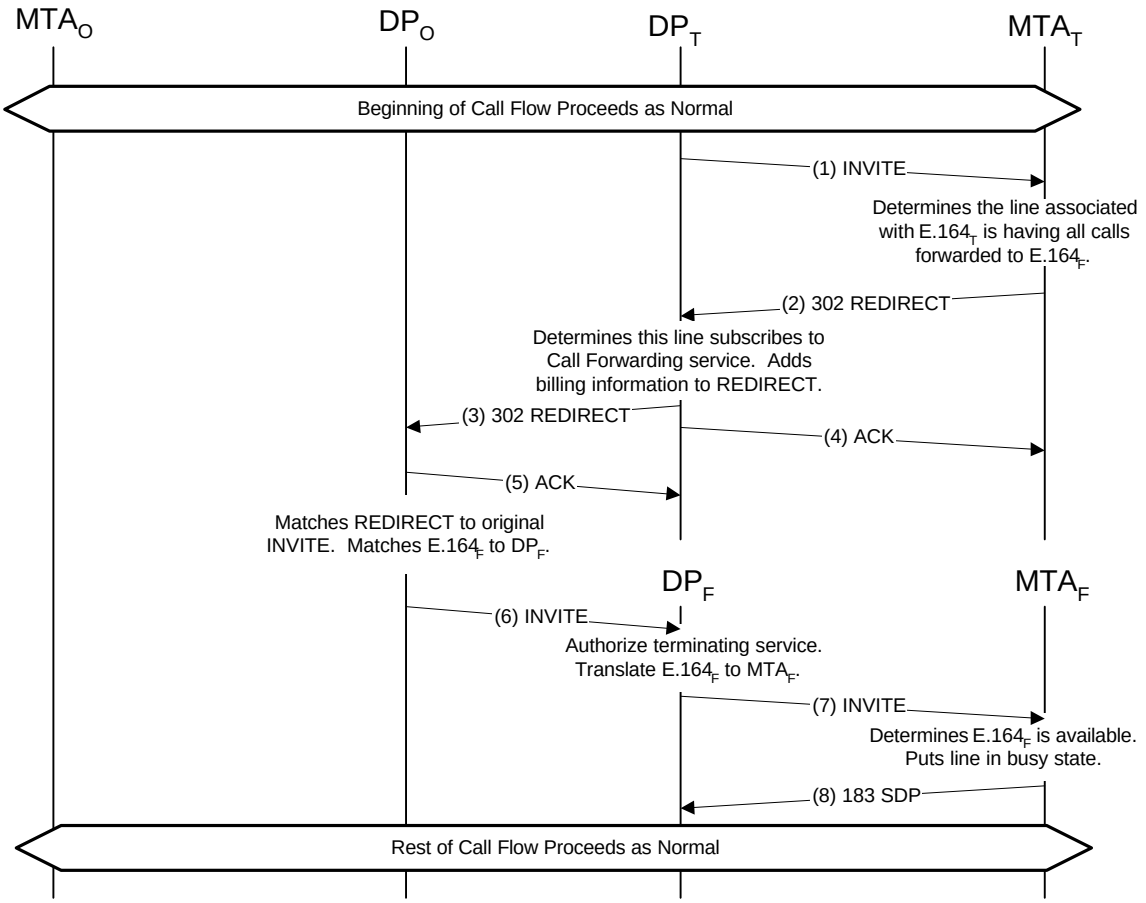


Figure 33: Call Forwarding Unconditional Signaling

The initial call flow for Call-Forwarding-Unconditional is identical to that shown in Figure 29 until MTA_T receives the following INVITE message from $CMS/Proxy_T$.

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA_T.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*
Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State blob encrypted with a $CMS/Proxy_T$ privately-held key containing: nexthop routing information, $CMTS_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this message, MTA_T determines that the line associated with 212-555-2222 is having all calls forwarded. It may initiate some local action (e.g. to play special ringing tones) to provide notification that the call is being forwarded. It may perform some functions as a SIP proxy, using the received Call-ID and SDP description, to further locate the user. It then issues a REDIRECT (302) response to indicate that it wants the call forwarded. This message carries the forwarding number in the Contact header.

(2) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as appear in INVITE message*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information requested by $CMS/Proxy_T$*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Call-ee information as provided by MTA*
Dcs-Anonymity: off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:555-3333	*New destination, contains the dialed digits from user initiating the service*

$CMS/Proxy_T$ verifies on receipt of the 302-Redirect message that the called party is a subscriber to the Call Forwarding service. $CMS/Proxy_T$ also verifies that the called party is permitted to forward the call to the supplied destination. It then adds a DCS-billing field to the 302-Redirect message to allow the "second leg" of the forwarded call to be charged to the user associated with 212-555-2222. It also restores the suppressed Via headers to allow the response to be routed back to $CMS/Proxy_O$.

(3) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(dp-o.provider); branch = 1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*Initial Billing Information, call originator to pay for first leg of forwarded call.*
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	*IP address and encryption key of the record keeping server for event collection: account number of called party/terminating number/forwarding number*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Billing Id*

Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
Dcs-Anonymity: off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:+1-212-555-3333	*New destination expanded to full E.164 number*

$CMS/Proxy_T$ also sends an ACK to MTA_T.

(4) ACK	***Description***
ACK sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Request-URI copied from initial INVITE*
Via: SIP/2.0/UDP Host(dp-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

The transaction at MTA_T is now complete.

$CMS/Proxy_O$ matches the 302 response to the INVITE it had sent out. It sends an ACK back to $CMS/Proxy_T$.

(5) ACK	***Description***
ACK sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Request-URI copied from initial INVITE*
Via: Sip/2.0/UDP Host(dp-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 127 ACK	

The transaction at $CMS/Proxy_T$ is now complete.

$CMS/Proxy_O$ determines the $CMS/Proxy_F$ for the E.164 number 212-555-3333 when it receives the 302-Redirect message. It generates an INVITE message and sends it to $CMS/Proxy_F$. It embeds two Dcs-Billing-Info headers in this message. The first one identifies the user associated with the E.164 number 212-555-1111 as paying for the initial call leg (212-555-1111/212-555-2222). The second one identifies the user associated with the E.164 number 212-555-2222 as paying for the second call leg (212-555-2222/212-555-3333). $CMS/Proxy_O$ adds the Dcs-Redirect header giving the information about this call redirection.

(6) INVITE:	***Description***
INVITE sip:+1-212-555-3333,lrn=212-265@Host(dp-f) ;user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(dp-o.provider); branch = 2	
Via: SIP/2.0/UDP Host(mta-o.provider);	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	*IP addr of CMTS, ID of the originating gate, and key for gate coord.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*Billing Info indicates split charging for this call: original caller is paying for a logical call from MTA_O to the initial destination, and the forwarding party is paying for a logical call from the initial destination to the final destination.*

Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_F$ to MTA$_O$*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS/Proxy*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_F$ queries the directory server to determine the IP address (MTA$_F$) associated with 212-555-3333. It then forwards the INVITE message to MTA$_F$.

(7) INVITE:	***Description***
INVITE sip:555-3333@Host(mta-f.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA$_F$.*
Via: SIP/2.0/UDP Host(dp-f.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*
Dcs-Media-Authorization: 22S21718	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-f.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-f.provider):4321/22S21718; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description of media stream to be received by MTA$_O$.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	

a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, MTA_F authenticates that the message came from $CMS/Proxy_F$ using IPSec. It checks the telephone line associated with the $E.164_F$ to see if it is available. If it is available, MTA_F looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA_F stores the INVITE message, including the encrypted Dcs-State parameters, for later use. MTA_F puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to $CMS/Proxy_F$. MTA_F starts timer (T-proxy-response).

(8) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-f.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as presented in INVITE message.*
Dcs-State: Host(dp-f.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-f.provider):4321/22S21718; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1"}$_K$"	*State information stored in MTA_F for this session.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	*Request for acknowledgement of this provisional response*
Session: qos	
Contact: sip:Host(mta-f.provider)	*Address for future direct signaling messages to MTA_F*
Content-Type: application/sdp	*The response to INVITE in SIP must contain the SDP description of the media stream to be sent to MTA_F.*
Content-length: (...)	
v=0	*SDP contains the MTA_T bearer channel IP address, and negotiated voice encoding parameters*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-f.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

The subsequent signaling call flows are identical to those shown in Figure 29.

Appendix G Call Forwarding Busy Call Flow

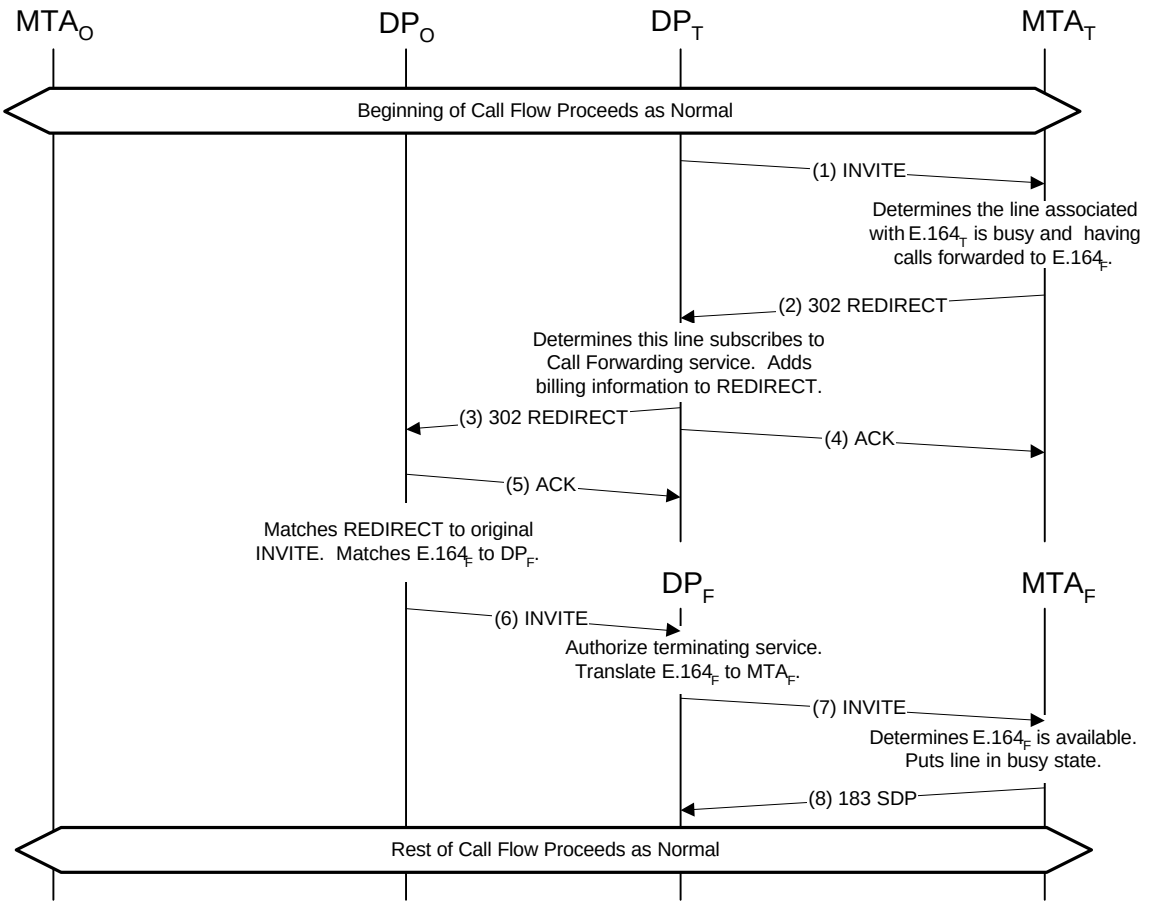


Figure 34: Call Forwarding Busy Signaling

The Call Forwarding Busy service is triggered when MTA_T detects that the called line ($E.164_T$) is busy. The subsequent call flow is identical to that for Call Forwarding Unconditional.

Note that the CMS/Proxy cannot reliably distinguish between Call Forwarding Unconditional and Call Forwarding Busy, and can therefore only offer these as a single service.

Appendix H Call Forwarding No-Answer Call Flow

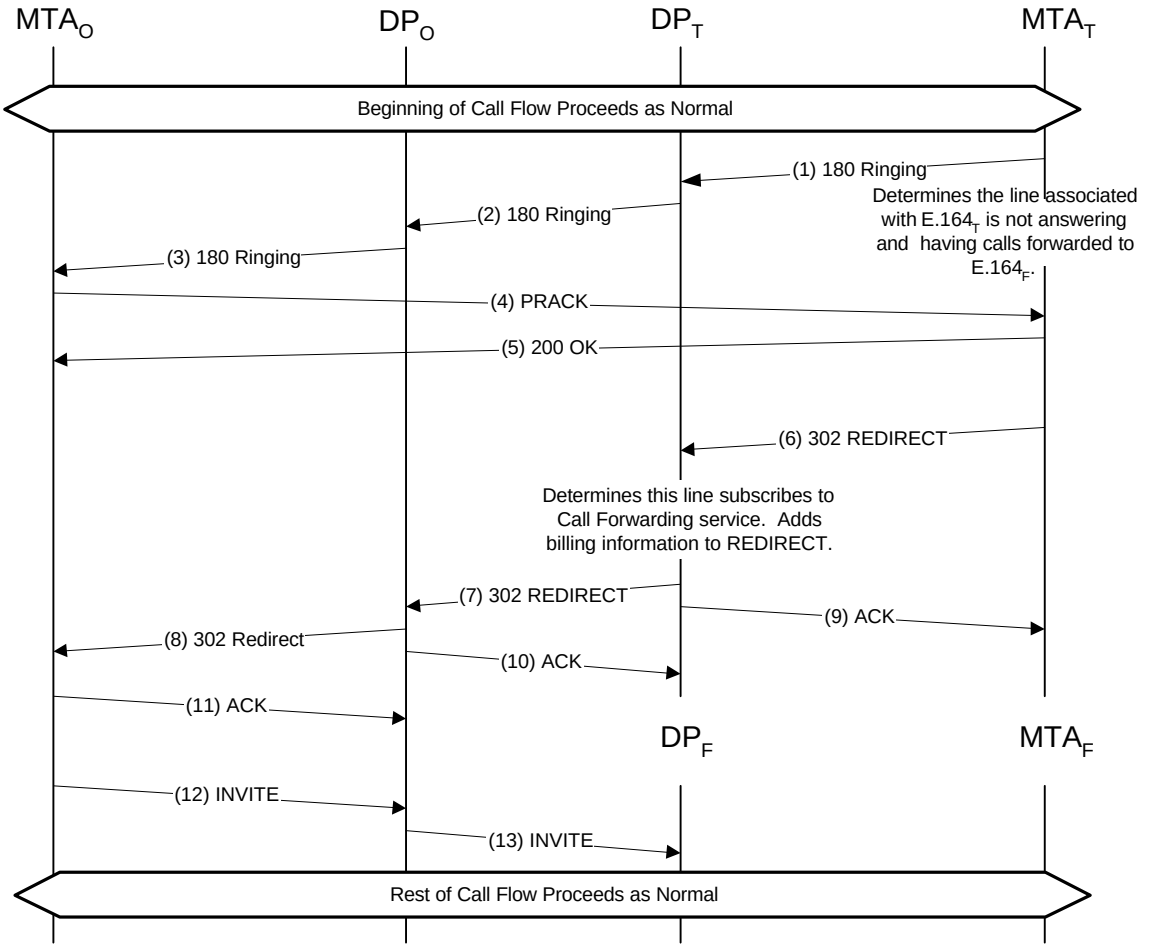


Figure 35: Call Forwarding No Answer Signaling

The Call Forwarding No Answer service is triggered when a called party does not pick up the phone after it rings for a pre-specified period of time. The subsequent call flow is different from that for the Call Forwarding Busy and Call Forwarding Unconditional services since the originating and terminating MTAs have already identified each other, have already reserved the resources for the call, and since the CMS/Proxies are no longer storing transaction state when the Forwarding function is triggered.

The initial set of messages for this service are the same as in Figure 29 through the point at which MTA_T is ringing the phone, and MTA_O is generating ringback.. For purposes of this example, consider the initial INVITE message received by MTA_T to be the following.

(not shown) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA_T.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*

Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cms-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cms-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA$_O$.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

In response to the INVITE message, MTA$_T$ starts local ringback and sends a 180 RINGING notification to MTA$_O$. It also starts the timer (T-ringing).

(1) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cms-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cms-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored for CMS/Proxy$_T$*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-o.provider)	
Cseq: 127 INVITE	
Rseq: 9022	*Request for acknowledgement of this provisional response*

The 180-Ringing message from CMS/Proxy$_T$ to CMS/Proxy$_O$ (2), the 180-Ringing message from CMS/Proxy$_O$ to MTA$_O$ (3), and the PRACK exchange (4) and (5), are identical to the basic call flow in Figure 29, and not repeated here.

When the timer(T-ringing) at the MTA$_T$ expires, it determines the forwarding number (555-3333) and sends a 302-Redirect response with this number in the Contact header.

(6) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	

Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as appear in INVITE message*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information requested by CMS/Proxy$_T$*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:555-3333	*New destination, contains the dialed digits from user initiating the service*

CMS/Proxy$_T$ uses its IPSec association with MTA$_T$ to determine the identity of the request. It then verifies the line subscribes to the Call-Forwarding-No-Answer service. CMS/Proxy$_T$ uses its Dcs-State value to recover the billing information for the current call (which is either stored directly in the Dcs-State value, or stored indirectly with a pointer to the Gate which stores the billing information). CMS/Proxy$_T$ adds an additional Dcs-Billing-Info header containing the billing information for the second leg of the forwarded call. CMS/Proxy$_T$ converts the new destination number in the Contact header into a full E.164 number, and passes the 302-Redirect message to CMS/Proxy$_O$.

(7) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(dp-o.provider); branch = 1	*Via headers as appear in INVITE message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*Original billing information, recovered from the gate parameters in this example.*
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	*Additional billing information for the second leg of the forwarded call.*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information requested by CMS/Proxy$_O$*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:+1-212-555-3333	*New destination, contains the dialed digits from user initiating the service*

CMS/Proxy$_O$ converts the Contact header into a private format URL containing the billing information and usage restrictions for the new call. By including a timestamp, CMS/Proxy$_O$ insures the URL can't be used for later call attempts beyond those authorized by the forwarder. Also encoded in the URL is the information needed for the Dcs-Redirect header and any required LAES.

(8) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Contact: sip:{type=transfer; dest=tel:+1-212-555-3333; billing-info= Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info= Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; billing-id= Host(dp-o.provider):36123E5C:0152; expires=36123E9A; orig-dest=+1-212-555-2222; redirected-by=+1-212-555-2222; num-redirects=1}$_K$@Host(dp-o.provider);private	*New destination, contains the dialed digits from user initiating the service*

CMS/Proxy$_T$ sends the following ACK message to MTA$_T$ after sending 302-Redirect(7).

(9) ACK	***Description***
ACK sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Request-URI copy of initial INVITE*
Via: SIP/2.0/UDP Host(dp-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

CMS/Proxy$_O$ sends the following ACK message to CMS/Proxy$_T$ after sending 302-Redirect(8).

(10) ACK	***Description***
ACK sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Reqquest-URI copy of initial INVITE*
Via: SIP/2.0/UDP Host(dp-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

MTA$_O$ sends the following ACK message to CMS/Proxy$_O$ on receipt of the 302-Redirect(8).

(11) ACK	***Description***
ACK sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Request-URI copy of initial INVITE*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

The transaction at CMS/Proxy$_O$, CMS/Proxy$_T$ and MTA$_T$ is now complete.

MTA$_O$, if it so desires, may now initiate a new call to the destination given in the Contact header. To avoid confusion at MTA$_O$, the call leg identification for this new call is different from that of the previous call. Therefore, any stored Dcs-State headers are not included in this INVITE, and only the Request-URI gives the handling and billing information.

(12) INVITE:	***Description***
INVITE sip:{type=transfer; dest=tel:+1-212-555-3333; billing-info= Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info= Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; billing-id= Host(dp-o.provider):36123E5C:0152; expires=36123E9A; orig-dest=+1-212-555-2222; redirected-by=+1-212-555-2222; num-redirects=1}$_K$@Host(dp-o.provider);private SIP/2.0	*Request URI taken from the Contact header of 302-Redirect*
Via: SIP/2.0/UDP Host(mta-o.provider)	*IP Address or Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	*Calling name and number, as provided by MTA*
Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*

From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	*Call leg identification is different from the previous call, in at least one component.*
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-o.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (...)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuite:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc- codecs:96	

CMS/Proxy$_O$ does all its normal authorization and authentication functions, and decodes the encrypted private username in the Request-URI. From that it builds the Dcs-Billing-Info, Dcs-Billing-ID, and Dcs-Redirect headers, and determines the destination address. The INVITE message sent on to CMS/Proxy$_F$ is as follows.

(13) INVITE:	***Description***
INVITE sip:+1-212-555-3333,lrn=212-265@Host(dp-f);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(dp-o.provider);branch=2	*CMS/Proxy$_O$ IP address; branch indicates this is the second destination attempt*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/3S73916/518C3B22 required	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis required for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*Original billing information for the first leg of the forwarded call*
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	*Additional billing information for the second leg of the forwarded call.*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/3S73916; orig-dest=tel:+1-212-555-2222; num-redirects=1	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Billing-ID: Host(dp-o.provider):36123E98:0171	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Suggested encryption key inserted by CMS/Proxy-o*

a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

The remainder of the call procedes as in Figure 29.

Appendix I Call Forwarding With Network Registration

Figure 36: Call Forwarding Network Registration

MTA_T recognizes that the customer dialed the code to activate Call Forwarding, and prompts the customer for the forwarding telephone number. This information is sent to the CMS/Proxy in a REGISTER message.

(1) REGISTER	***Description***
REGISTER sip:Host(dp-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: sip:555-2222@Host(mta-t.provider); user=phone	*From: header contains unencrypted phone number*
To: sip:Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-2222;time=361013B8;seq=1))	
Cseq: 1 REGISTER	
Contact: tel:555-3333	*Contact contains URI of the desired forwarding destination*
Expires: 7200	

The CMS/Proxy validates that the forwarding number maps to either a MTA it knows about or to another valid CMS/Proxy. The CMS/Proxy also checks to make sure that the customer subscribes to the Call Forwarding service, and if so activates the service and stores the forwarding number for later use. It responds to the MTA with a 200-OK.

(2) **200-OK**	***Description***
SIP 2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: sip:555-2222@Host(mta-t.provider); user=phone	
To: sip:Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-2222;time=361013B8;seq=1))	
Cseq: 1 REGISTER	

Appendix J Call Forwarding MTA Unavailable Call Flow

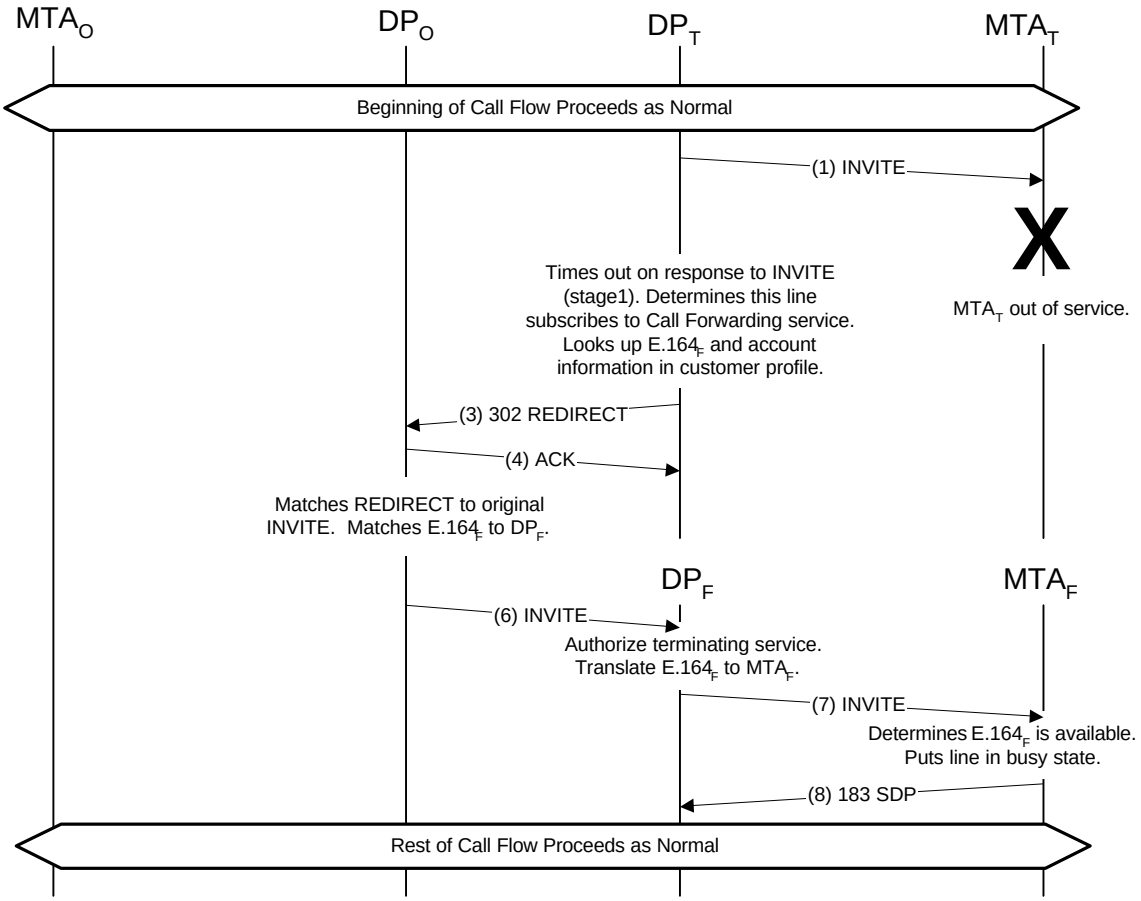


Figure 37: Call Forwarding when MTA unavailable

The initial sequence of messages is identical to that for a regular call setup as shown in Figure 29 until CMS/Proxy$_T$ forwards the INVITE message to MTA$_T$. CMS/Proxy$_T$ times out when it does not receive a response to the INVITE from MTA$_T$. It determines that the called party subscribes to Call Forwarding service and that the forwarding number is 212-555-3333. It then generates a SIP 302 (Redirect) message with the forwarding number in the Contact header. It then adds the DCS-billing and Dcs-Billing-ID fields to the 302 message that allows the second leg of the forwarded call to be charged to the user associated with 212-555-2222. The subsequent call flow is the same as with Call Forwarding Unconditional, (or Call Forwarding Busy), and is given in Appendix F.

Appendix K Return-Call Service

We assume for this example that MTA_T had last received a call from MTA_O. The INVITE message forwarded to MTA_O included the Dcs-Remote-Party-ID line, which contained, among other items, a URL that identified MTA_O. If the original caller did not request privacy, and the destination subscribed to caller-id, then the URL contains the E.164 number, which can be used to place the return call. We assume for this example that was not the case, and that MTA_T does not know the identity of the new call's destination.

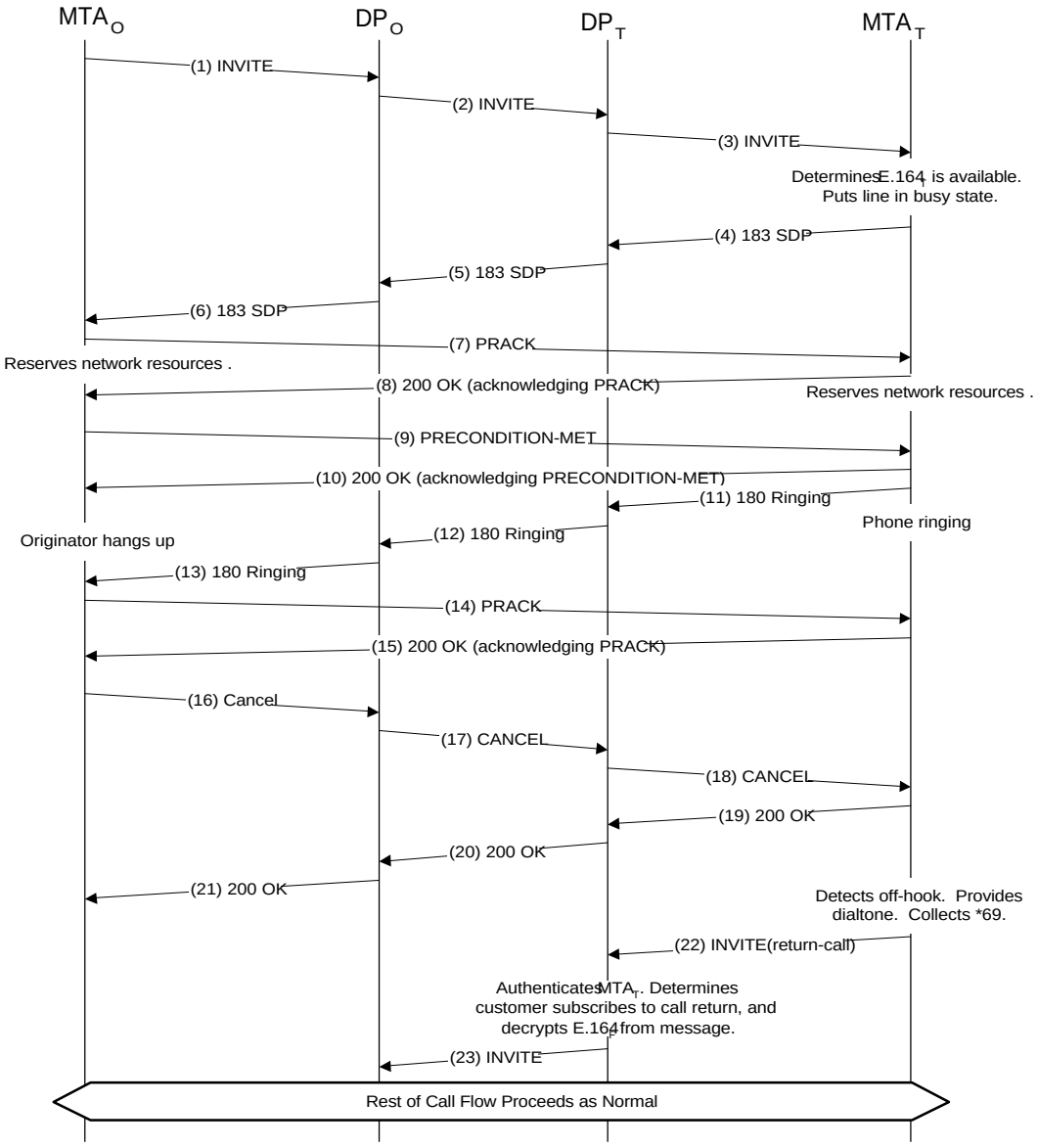


Figure 38: Return Call Signaling

Messages (1) through (15) in the above diagram are identical to those for the basic call flow given in Figure 29, and message (16) through (21) is a standard SIP CANCEL operation. The key parameters used in processing the return-call are contained in message (3), reproduced below. For purposes of this example,

we assume the destination had not subscribed to Caller-ID service, and therefore the calling-name and calling-number information is not present in (3) INVITE.

(3) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Username is a string known to MTA_T.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: <sip:{type=remote-id; orig=tel:+1-212-555-1111; otherstuff=whatever}$_K$@Host(dp-t.provider); private>; rpi-id=na	*Without the Caller-ID and Calling-Name service, the Remote-party-ID header contains a private URL that identifies the caller to the trust network elements only.*
Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon the user dialing *69, MTA_T initiates a call by sending an INVITE message to its CMS/Proxy, with the Request-URI containing the URL for the call to be returned. The complete message is as follows.

(22) INVITE:	***Description***
INVITE sip:{type=remote-id; orig=tel:+1-212-555-1111; otherstuff=whatever}$_K$@Host(dp-t.provider); private SIP/2.0	*Request URI contains the URL of the caller, as given in the Remote-Party-ID header of the previous call*
Via: SIP/2.0/UDP Host(mta-t.provider)	*Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Originator name and number supplied by MTA*
Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*
From: sip:B64(SHA-1(555-2222; time=36123F12;seq=3))@localhost	*The triple (From, To, CallID) uniquely identifies the call at the two endpoints. To maintain privacy, the Originating Name and Number are encrypted with originator's key. Call-ID is a (hopefully) unique ASCII encoding of a random number*
To: sip:B64(SHA-1(*69; time=36123F12;seq=4))@localhost	
Call-ID: B64(SHA-1(555-2222;time=36123F12;seq=3))@localhost	
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-t.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*

o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 7242 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving the INVITE message, CMS/Proxy$_T$ authenticates MTA$_T$ using standard IPSec. CMS/Proxy$_T$ decrypts the destination string using its privately-held key, and checks its signature in the result. From this string the real destination E.164 is extracted. CMS/Proxy$_T$ checks the "Dcs-Remote-Party-ID:" line, and checks to see that this line belongs to MTA$_T$, and has either subscribed to call-return service, or is authorized to use the service and be charged on a per-use basis. CMS/Proxy$_T$ then performs all the regular call handling functions, as described in the basic call flow. The message sent to CMS/Proxy$_O$ is the following, and the call proceeds identically to the basic call flow from this point onward.

(23) INVITE:	***Description***
INVITE sip:+1-212-555-1111,lrn=212-237@Host(dp-o.provider);user=np-queried SIP/2.0	*"lnp=212-237" shows that LNP dip done and LRN*
Via: SIP/2.0/UDP Host(dp-t.provider);branch=1	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Via: SIP/2.0/UDP Host(mta-t.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	*Verified Caller-name and Caller-number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	*IP address of the originating gate (the originating CMTS)*
Dcs-Billing-Info: Host(rks-t.provider)<5098-0987-6543-2100/212-555-2222/212-555-1111/*69>	*IP address and encryption key of the record keeping server for event collection, Account number/originating number/terminating number for billing, also an indication of special services to be charged*
Dcs-Billing-ID: Host(dp-t.provider):36123F12:0381	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
Dcs-State: Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621	*State information wanted by CMS/Proxy$_T$ for handling messages from CMS/Proxy$_O$ to CMS/Proxy$_T$.*
From: sip:B64(SHA-1(555-2222; time=36123F12;seq=3))@localhost	*The triple (From, To, CallID) is used by SIP to uniquely identify a call*
To: sip:B64(SHA-1(*69; time=36123F12;seq=4))@localhost	
Call-ID: B64(SHA-1(555-2222;time=36123F12;seq=3))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Key provided by CMS/Proxy for this connection*
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 7242 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Remainder of call proceeds identically to the basic call flow given in Figure 29

Appendix L Customer Originated Trace Call Flow

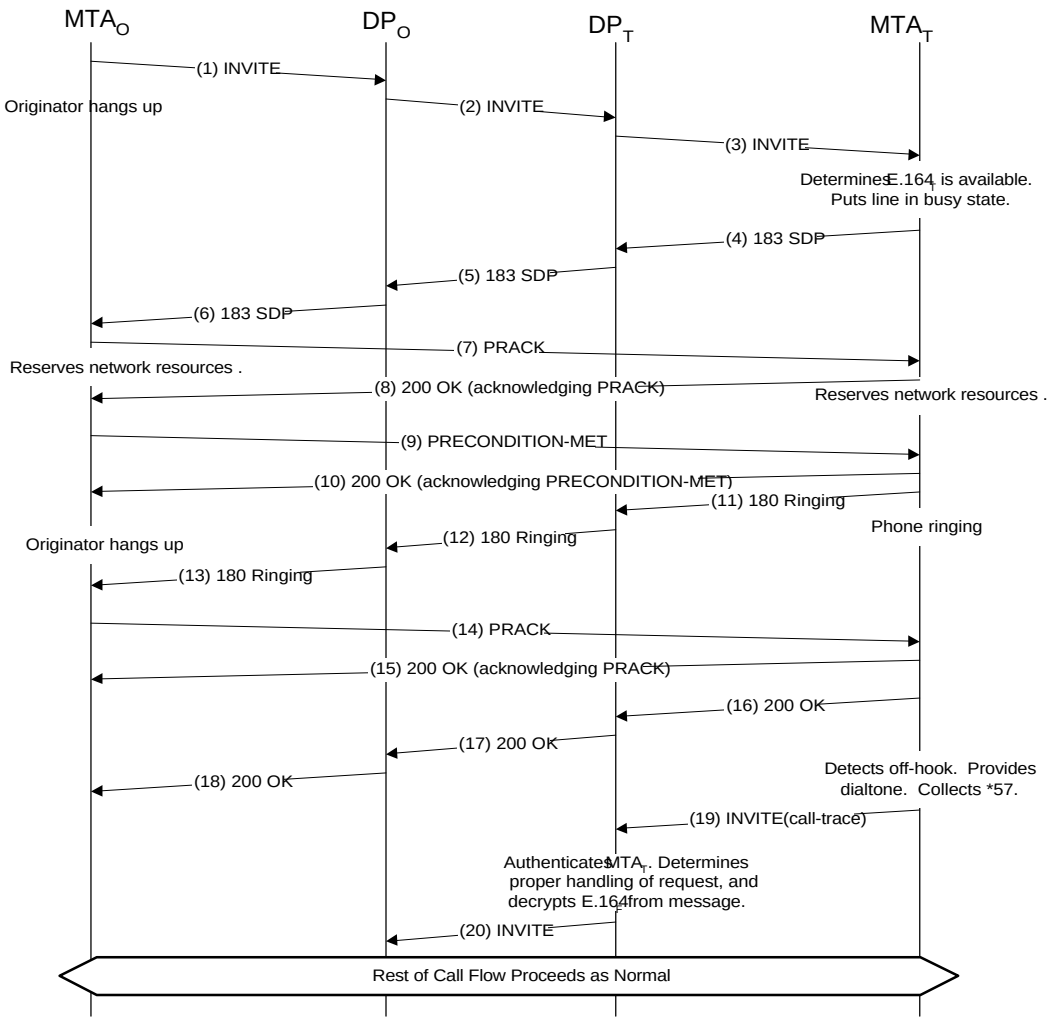


Figure 39: Call Trace Signaling

Design of call-trace (*57) is almost identical to return-call (*69), but the action taken by the CMS/Proxy is to report the information to law enforcement authorities, and complete the call either to the Service Provider's office or to an announcement server (which tells the customer to call the Service Provider's office).

(19) INVITE:	***Description***
INVITE sip:call-trace@Host(dp-t.provider) SIP/2.0	*Request URI contains the reserved destination "call-trace"*
Via: SIP/2.0/UDP Host(mta-t.provider)	*Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Trace-Party-ID: sip:{type=remote-id; orig=tel:+1-212-555-1111; otherstuff=whatever}$_K$@Host(dp-t.provider); private	*Identity of caller being repported, from the Remote-Party-ID of the previous call*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Originator name and number supplied by MTA*

Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*
From: sip:B64(SHA-1(555-2222; time=36123F12;seq=3))@localhost	*The triple (From, To, CallID) uniquely identifies the call at the two endpoints. To maintain privacy, the Originating Name and Number are encrypted with originator's key. Call-ID is a (hopefully) unique ASCII encoding of a random number*
To: sip:B64(SHA-1(*57; time=36123F12;seq=4))@localhost	
Call-ID: B64(SHA-1(555-2222;time=36123F12;seq=3))@localhost	
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-t.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 7242 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

CMS/Proxy performs the reporting function and connects to either (1) an announcement server telling customer the information is recorded, and to now call the Business Office during normal business hours, or (2) the Business Office.

Appendix M Call Waiting Call Flow

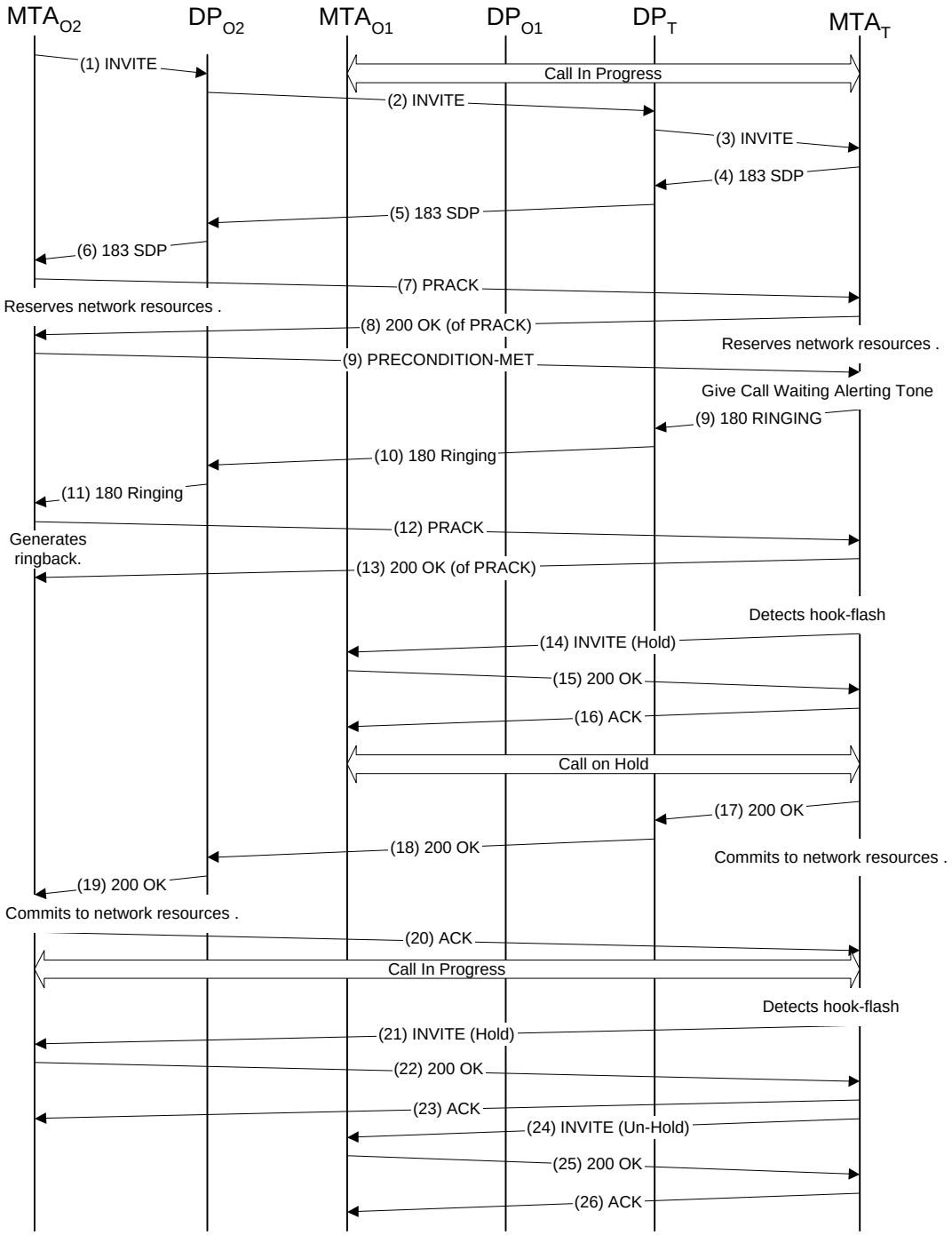


Figure 40: Call Waiting – Signaling Flow

Call Waiting is a service that allows a customer to respond to an incoming call during the time the phone line is busy. The customer hears an audible alerting tone, and indicates acceptance of the new call via a hookflash (putting the previous call on hold). Subsequent hookflashes switch between the two active calls. The originator of the second call MAY hear a distinctive ringback tone.

For this example, consider an existing call initiated by MTA_{O1}, with the following call identification:

MTA_T state for call from MTA_{O1} to MTA_T	***Description***
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	Call leg identification
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-o1.provider)	Contact address for end-to-end signaling messages
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o1.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o1.provider); nexthop=sip:555-1111@Host(mta-o1.provider); gate=Host(cmts-o1.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	Encrypted state information from CMS/Proxy stored in MTA
Dcs-Billing-Info: Host(rks-o1.provider)/04FA37<5123-0123-4567-8900/212-555-1111/212-555-2222>	Billing Information, stored in Gate
Dcs-Billing-ID: Host(dp-o1.provider):36123E5C:0152	Unique Billing identifier for this call

The initial set of messages associated with the second arriving call, (1) through (13), as shown in Figure 40, are very similar to those involved in a Basic Call Setup and are not explicitly enumerated below. After the initial INVITE exchange, the state information stored for this new call is:

MTA_T state for call from MTA_{O2} to MTA_T	***Description***
From: sip:B64(SHA-1(555-3333; time=36124125;seq=23))@localhost	Call leg identification
To: sip:B64(SHA-1(555-2222; time=36124125;seq=24)@localhost	
Call-ID: B64(SHA-1(555-3333;time=36124125;seq=23))@localhost	
Contact: sip:Host(mta-o2.provider)	Contact address for end-to-end signaling messages
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o2.provider); gate=Host(cmts-t.provider):4321/32S35378; state="Host(dp-o2.provider); nexthop=sip:555-3333@Host(mta-o2.provider); gate=Host(cmts-o2.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	Encrypted state information from CMS/Proxy stored in MTA
Dcs-Billing-Info: Host(rks-o2.provider)/173F419B<6010-4500-6789-0123/212-555-3333/212-555-2222>	Billing Information, stored in Gate
Dcs-Billing-ID: Host(dp-o2.provider):36124125:0031	Unique Billing identifier for this call

In response to the INVITE for the second incoming call, the user at MTA_T is provided some indication of the second call, e.g. using a special tone. If the user at MTA_T hits a flash hook in response to this, MTA_T issues a INVITE(Hold) message to MTA_{O1} to put it on HOLD.

(14) INVITE (Hold):	***Description***
INVITE sip:Host(mta-o1.provider) SIP/2.0	*Address from Contact header of initial INVITE or initial 183*
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 INVITE	

(16) ACK	***Description***
ACK Host(mta-o1.provider)	*Address from Contact header of initial INVITE, or from 183*
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 ACK	

Once the first conversation is successfully placed on hold, MTA_T indicates a completion to the "ringing" to MTA_{O2}.

(17) 200-OK	*Description*
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o2.provider); gate=Host(cmts-t.provider):4321/32S35378; state="Host(dp-o2.provider); nexthop=sip:555-3333@Host(mta-o2.provider); gate=Host(cmts-o2.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*Encrypted state information from CMS/Proxy stored in MTA*
From: sip:B64(SHA-1(555-3333;time=36124125;seq=23))@localhost	*Call leg identification of second call.*
To: sip:B64(SHA-1(555-1111; time=36124125;seq=24)@localhost	
Call-ID: B64(SHA-1(555-3333;time=36124125;seq=23))@localhost	
Cseq: 128 INVITE	

This 200-OK is passed through the proxy chain in messages (18) and (19) to MTA_{O2}. MTA_{O2} responds with an acknowledgement, in a manor identical to the basic call flow.

(20) ACK	*Description*
ACK Host(mta-t.provider)	*Address from Contact header of original INVITE message*
Via: SIP/2.0/UDP Host(mta-o2.provider)	
From: sip:B64(SHA-1(555-3333;time=36124125;seq=23))@localhost	
To: sip:B64(SHA-1(555-1111; time=36124125;seq=24)@localhost	
Call-ID: B64(SHA-1(555-3333;time=36124125;seq=23))@localhost	
CSeq: 128 ACK	

At this point the user at MTA_T has a connection to the second caller, MTA_{O2}, with the first caller, MTA_{O1}, on hold.

Subsequent hookflashes repeat the sequence of INVITE(hold)/200-OK/ACK to one destination, and INVITE(resume)/200-OK/ACK to the other. The INVITE (Hold) sequence (15) through (17) is identical to (10) through (12). Once the 200-OK is received, it is safe for MTA_T to stop sending voice packets.

INVITE (Resume) is very similar, except that the SDP description includes the proper IP address in the "c=" line.

(24) INVITE (Resume):	*Description*
INVITE sip:Host(mta-o1.provider) SIP/2.0	*Address from Contact header of initial INVITE or 200-OK*
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 130 INVITE	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description. The real address tells MTA_{O1} to take this line off hold*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

MTA_{O1} acknowledges the Resume command with a 200-OK message. The response contains an updated SDP description for the stream to be received at MTA_{O1}, indicating the real IP address of Host(mta-o1.provider).

(25) 200-OK	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*Updated SDP description, showing normal call.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o1.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	

MTA_T responds to the 200-OK message with the standard SIP ACK message. At this point it is safe for MTA_T to start sending voice payload packets to MTA_{O1}.

(26) ACK	***Description***
ACK Host(mta-o.provider)	*Address from Contact header of Initial INVITE or initial 183*
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 ACK	

Appendix N Call Transfer (Blind) Call Flow

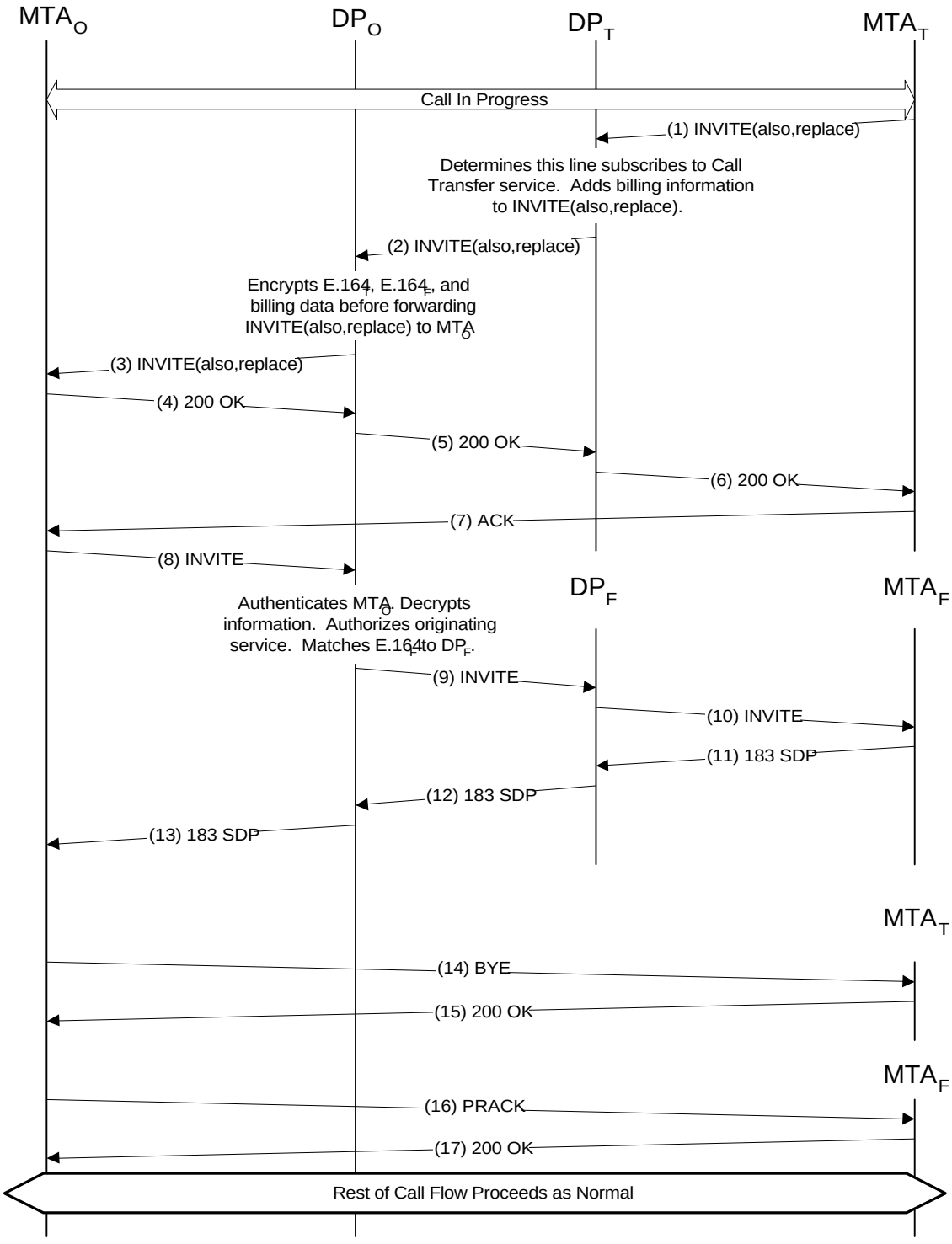


Figure 41: Call Transfer Signaling

The Call Transfer service is triggered by the user by methods beyond the scope of this specification. Described in this section is a transfer service common known as "blind transfer" where the party initiating the transfer (MTA_T in this example) is not informed of the success or failure of the transfer operation. The alternative, commonly known as "consultative transfer" is described later.

For this example, consider an existing call initiated by MTA_O, with the following call identification:

MTA_T state for call from MTA_O to MTA_T	***Description***
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	Call leg identification
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-o.provider)	Contact address for end-to-end signaling messages
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	Encrypted state information from CMS/Proxy stored in MTA
Dcs-Billing-Info: Host(rks-o.provider)/04FA37<5123-0123-4567-8900/212-555-1111/212-555-2222>	Billing Information, stored in Gate
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	Unique Billing identifier for this call

When MTA_T desires to transfer the existing call, it determines the forwarding number (in this example 555-3333) and issues an INVITE(also,replace) message to MTA_O. INVITE(also,replace) is the same as a regular INVITE but includes an additional "Dcs-Also:" header and "Dcs-Replaces:" header. The "Dcs-Also:" header identifies the number to which the call needs to be forwarded, while the "Dcs-Replaces:" header identifies the existing call leg at MTA_O. The following message is sent to MTA_T's CMS/Proxy, $CMS/Proxy_T$.

(1) INVITE(also,replace):	***Description***
INVITE sip: Host(mta-o.provider) SIP/2.0	*Request URI contains the value from the Contact header of the initial INVITE.*
Via: SIP/2.0/UDP Host(mta-t.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Also: tel:555-3333	*Identifies new call leg to be created*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*Encrypted state information from CMS/Proxy stored in MTA*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	
Dcs-Anonymity: off	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*From: is copied from the original To:*
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*To: is copied from the original From:*
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	*Call-ID is kept identical to original*
Cseq: 8001 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*Dcs-Replaces: is copied from the original To:*

When the INVITE(also,replace) is received at $CMS/Proxy_T$, it first verifies MTA_T has subscribed to Call Forwarding service. If so, it decrypts the Dcs-State information to determine the local gate location and identification. $CMS/Proxy_T$ queries the gate to obtain the call's billing information. $CMS/Proxy_T$ inserts billing information to indicate that the user associated with the number 212-555-2222 will pay for the new

call segment. CMS/Proxy$_T$ extracts the call routing from the Dcs-state information, and then forwards the message to CMS/Proxy$_o$.

(2) INVITE(also,replace):	***Description***
INVITE sip: Host(dp-o.provider) SIP/2.0	*Request-URI obtained from Dcs-State information*
Via: SIP/2.0/UDP Host(dp-t.provider)	
Via: SIP/2.0/UDP Host(mta-t.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information as requested by CMS/Proxy$_O$*
Dcs-Also: tel:+1-212-555-3333? Dcs-Billing-Info= Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222> & Dcs-Billing-Info= Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333> & Dcs-Billing-ID= Host(dp-o.provider): 36123E5C:0152	*Expanded to full E.164 number. Original billing information will be used for a pseudo-call from originator to the point where the call was forwarded, and new billing information used for a pseudo-call from forwarding location to the new destination. Original billing identifier is kept for the forwarded call*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*Call leg identification*
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

CMS/Proxy$_o$ forwards the INVITE(also,replace) message to MTA $_O$ after encrypting the <Dcs-Billing, Dcs-Billing-ID> headers.

(3) INVITE(also,replace):	***Description***
INVITE sip: 555-1111@Host(mta-o.provider) SIP/2.0	*Request-URI obtained from Dcs-State information*
Via: SIP/2.0/UDP Host(dp-o.provider), {via="Host(dp-t.provider); branch=1"; via=Host(mta-t.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Also: sip:{type=transfer; dest=tel:+1-212-555-3333; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info=Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info=Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; orig-dest=tel:+1-212-555-2222; redirected-by=tel:+1-212-555-2222; num-redirects=1}$_K$@Host(dp-o.provider);private	*Dcs-Also: contains the encrypted forwarder, new destination, Billing-identifier, timestamp, and Billing-Information fields. All are checksummed, signed by CMS/Proxy$_O$, and encrypted.*
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*Call leg identification*
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

MTA$_O$ acknowledges receipt of the INVITE(also,replace) by sending a 200-OK to MTA$_T$. This message is routed through the CMS/Proxy CMS/Proxy$_O$, CMS/Proxy$_T$, and then delivered to MTA$_T$. MTA$_T$ responds directly with an ACK. CMS/Proxy$_T$ is now done, while MTA$_T$ is waiting for the BYE message, which will come after MTA$_O$ contacts the new destination.

(4) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider), {via="Host(dp-t.provider); branch=1"; via=Host(mta-t.provider)}$_K$	*Via headers, as given in the INVITE message*

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222,lrn=212-234@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	

CMS/Proxy$_O$ restores the encrypted Via headers, and forwards the OK to topmost Via – CMS/Proxy$_T$.

(5) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider)	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	

CMS/Proxy$_T$ forwards the 200-OK to MTA$_T$.

(6) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	

MTA$_T$ responds with an ACK message.

(7) ACK:	***Description***
ACK sip:Host(mta-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mta-t.provider)	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 ACK	

After processing the INVITE(also,replace), MTA$_O$ issues a INVITE to MTA$_F$. In addition to the standard headers carried in an INVITE message, the encrypted {Dcs-Billing, Dcs-Billing-ID} fields received in the INVITE(also,replace) message are copied into the INVITE message. These fields indicate that the user associated with the 212-555-2222 number will be charged for the second call leg.

(8) INVITE:	***Description***
INVITE sip:{type=transfer; dest=tel:+1-212-555-3333; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info=Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info=Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; orig-dest=tel:+1-212-555-2222; redirected-by=tel:+1-212-555-2222; num-redirects=1}$_K$@Host(dp-o.provider);private SIP/2.0	*Destination for the INVITE is taken from the Dcs-Also: header in the INVITE-REPLACE above. Private-param indicates the information is encrypted, and the first encrypted item, transfer, indicates the format.*
Via: SIP/2.0/UDP Host(mta-o.provider)	

Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	*Originator supplied calling name and number*
Dcs-Anonymity: Off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	*Call leg identification information. Note that MTA_O does not know the identity of the new destination, and therefore retains the previous To: header. Sequence number in SHA-1 hash incremented.*
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 129 INVITE	
Contact: sip:Host(mta-o.provider)	*Local (non-NAT) address for further end-to-end signaling exchanges*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

When the $CMS/Proxy_O$ receives the INVITE it first decrypts the header information to find the real destination for the call. CMS/Proxy compares the current time against the timestamp in the encrypted string; if the request is too old, it is refused. It invokes the call routing logic to determine which CMS/Proxy ($CMS/Proxy_F$) to which the INVITE needs to be routed. It also embeds two Dcs-Billing-Info headers in this message. The first one identifies the user associated with the E.164 number 212-555-1111 as paying for the initial call leg (212-555-1111/212-555-2222). This information was derived from the customer account information for the caller during the first call attempt. The second Dcs-Billing-Info header identifies the user associated with the E.164 number 212-555-2222 as paying for the second call leg (212-555-2222/212-555-3333), and was provided by $CMS/Proxy_T$ in the INVITE(also,replace) message.

(9) INVITE:	***Description***
INVITE sip: +1-212-555-3333,lrn=212-265@Host(dp-f);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(dp-o.provider); branch=1;	
Via: SIP/2.0/UDP Host(mta-o.provider);	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	*Verified Caller Identification, with full E.164 number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection: account number/originating number/terminating number for billing. From the URI in INVITE message*
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	*Further billing information regarding split charging. Information from the URI in INVITE message.*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of $CMS/Proxy_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 129 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	

s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_F$ queries the directory server to determine the IP address (MTA$_F$) associated with 212-555-3333. It then forwards the INVITE message to MTA$_F$, after stripping off all of the billing fields, and adding the encrypted state information. This is identical to the basic call flow shown in Figure 29, and is not repeated here.

Upon receipt of the 183-Session-Progress message, MTA$_o$ sends the usual acknowledgement (PRACK) message, and also sends the following BYE message to the original destination.

(14) BYE:	***Description***
BYE sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification of original call*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 BYE	

Upon receipt of the BYE message, MTA$_T$ releases all network resources that have been used for this call. MTA$_T$ sends the following 200-OK message to MTA$_O$.

(15) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 BYE	

Appendix O Call Transfer (Consultation) Call Flow

Call Transfer with Consultation is triggered by the user by methods beyond the scope of this specification. It consists of two distinct phases: first placing the existing call on hold and placing a new call to another destination (the consultation), and secondly transfering the first call to the second destination (the transfer).

For this example, consider an existing call initiated by MTA_{T1} to MTA_O. The call identification information at MTA_O is as follows:

MTA_O state for call from MTA_{T1} to MTA_O	***Description***
From: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	Call leg identification
To: tel:555-1111	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Contact: sip: Host(mta-t1.provider)	Contact address for end-to-end signaling messages
Dcs-Remote-Party-ID: tel:+1-212-555-2222	Identity of other party in this call
Dcs-State: Host(dp-o.provider); state="{nexthop=sip:Host(dp-t1.provider); gate=Host(cmts-o.provider):3612/17S30124; state="Host(dp-t1.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t1.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	State information of CMS/Proxy, encrypted and stored in MTA
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>	Billing information stored at gate
Dcs-Billing-ID: Host(dp-t1.provider):36124033:0381	Unique identifier for this call

MTA_O places this call on hold and determines the destination for consultation. MTA_O initiates a second call to the consultation endpoint, MTA_{T2}, as shown in Figure 42.

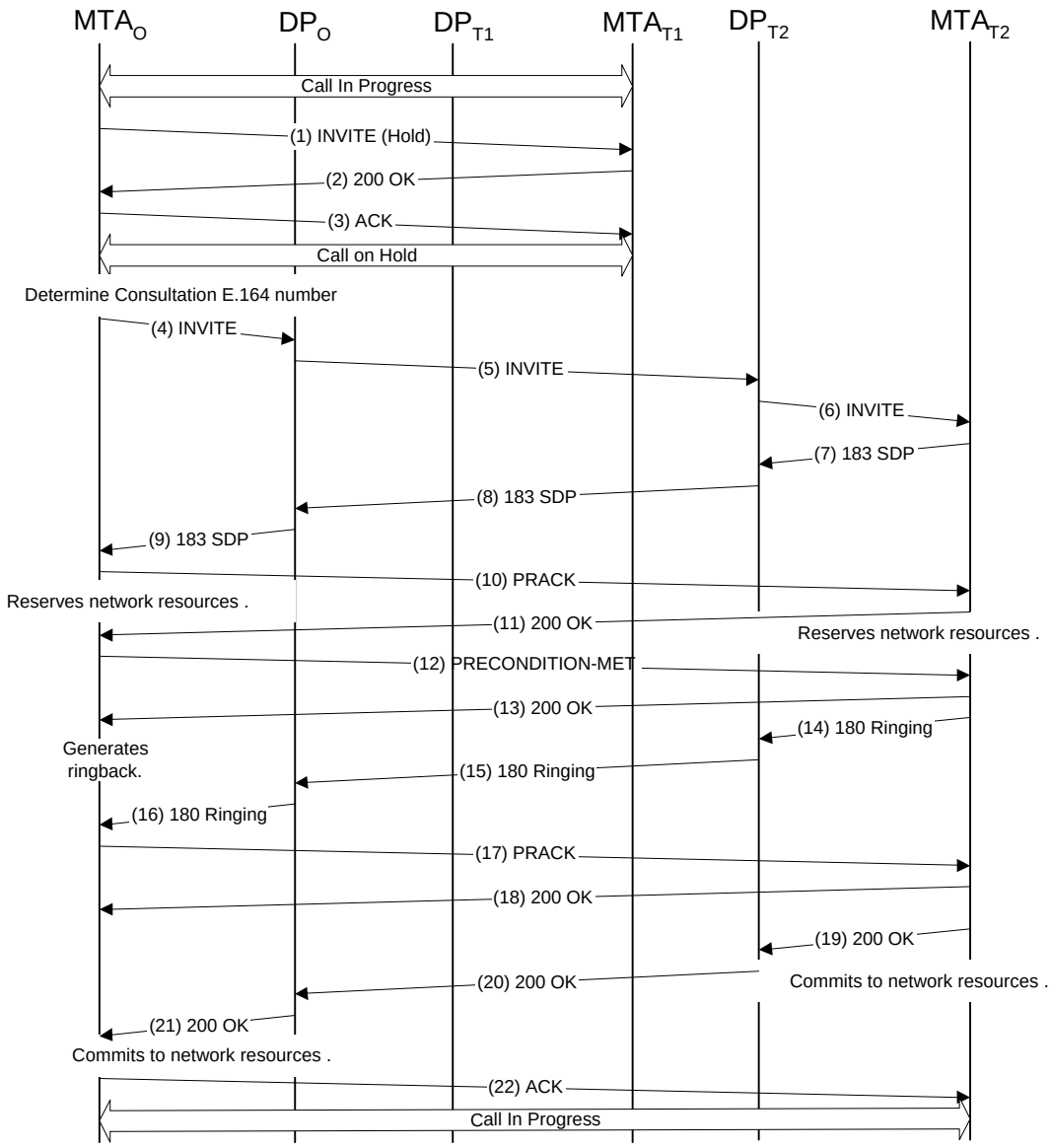


Figure 42: Call Transfer w/Consultation #1 - Consultation

Signaling messages (1) to (3), placing the first call on hold, are identical to those used in Call Waiting (see Figure 40), and are not reproduced here.

Signaling messages (4) to (22), placing the second call, are identical to those for a basic call flow (see Figure 29), and are not reproduced here. For this example, assume the Call-ID was B64(SHA-1(555-1111;time=36124125;seq=23))@localhost.

State at MTA_O for call from MTA_O to MTA_{T2}	***Description***
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	Call leg identification
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Contact: sip: Host(mta-t2.provider)	Contact address for end-to-end signaling messages
Dcs-Remote-Party-ID: tel:+1-212-555-3333	Identity of other party in this call

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/3S10782, nexthop=sip:+1-212-555-3333,lrn=212-256@Host(dp-t2.provider), state="Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	State information of CMS/Proxy, encrypted and stored in MTA
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>	Billing information stored at gate
Dcs-Billing-ID: Host(dp-o.provider):3612E5C:0152	Unique identifier for this call

After some period of consultation, MTA_O initiates a transfer of the call from MTA_{T1} to the new destination, MTA_{T2}. This involves placing the second call on hold (message sequence described earlier), and sending an INVITE(also,replace) message to MTA_{T2}, giving it the information about the call with MTA_{T1} in the Also: header. The INVITE message, since it changes parties involved in the call, is routed through the proxies. The sequence is shown in Figure 43, and detailed below.

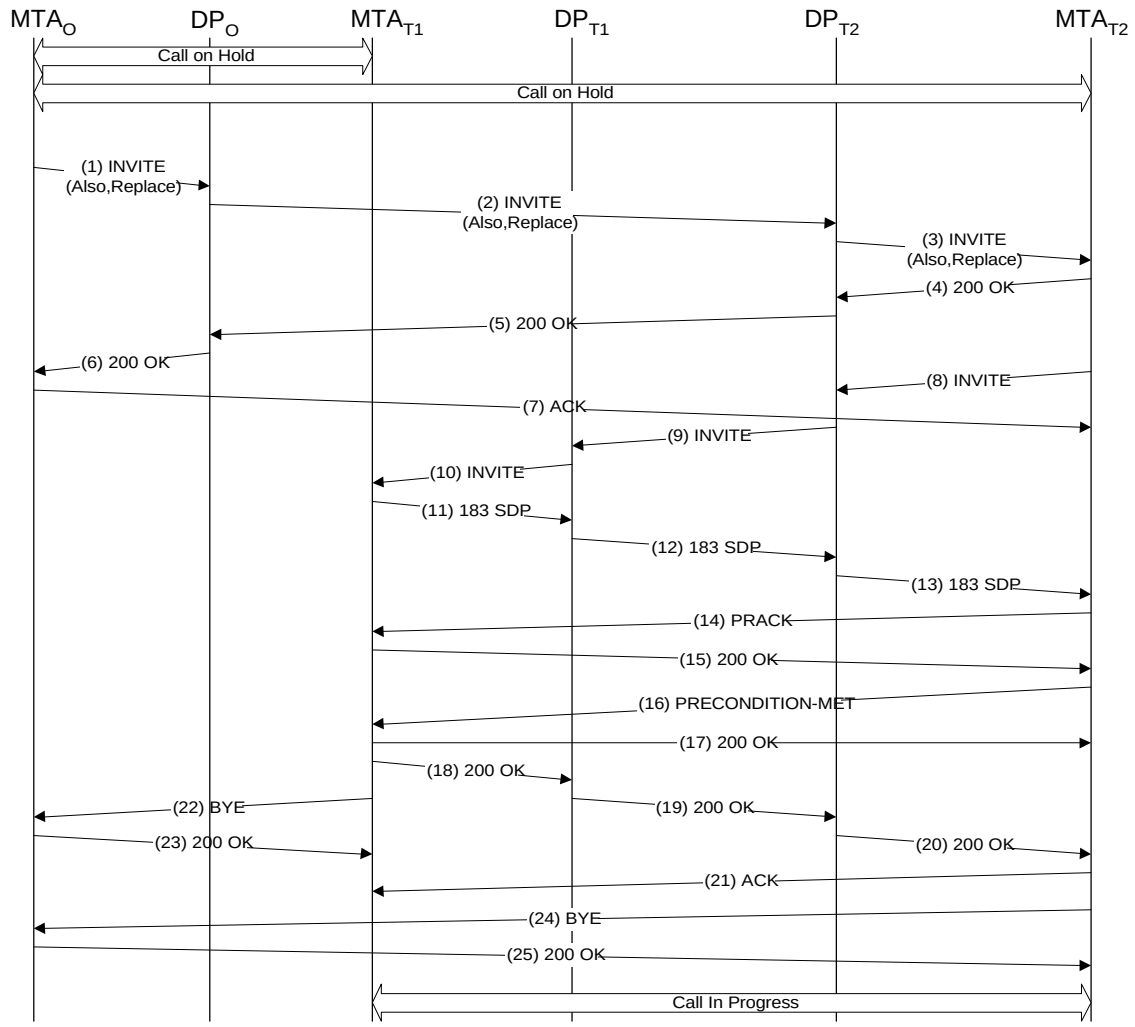


Figure 43: Call Transfer w/Consultation #2 - Transfer

After placing the second call on hold, MTA_O initiates a transfer by sending an INVITE(also,replace) to MTA_{T2}, routed through the proxies.

(1) INVITE(also,replace):	*Description*
INVITE sip: Host(mta-t2.provider) SIP/2.0	*Address from Contact header of previous INVITE*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
Dcs-Anonymity: off	
Dcs-Also: tel:+1-212-555-2222 ? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111 & Dcs-State= Host(dp-o.provider); state="{nexthop=sip:Host(dp-t1.provider); gate=Host(cmts-o.provider):3612/17S30124; state="Host(dp-t1.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t1.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*Identifies new call leg to be created. The URL is from the Dcs-Remote-Party-ID header of call from mta-t1. Call-ID gives the Call-ID to be used for the new call mta-t2 makes to mta-t1. Use of the same Call-ID as MTA$_O$'s call with mta-t1 causes the transfer. Dcs-State included due to matching Call-ID and Dcs-Replaces matching To: header value of call from mta-t1 to mta-o.*
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/3S10782, nexthop=sip:+1-212-555-3333,lrn=212-256@Host(dp-t2.provider), state="Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State information of CMS/Proxy, encrypted and stored in MTA*
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	*Call leg identification. Identifies call from mta-o to mta-t2*
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	*Dcs-Replaces identifies the existing call leg to be torn down*

When the INVITE(also,replace) is received at CMS/Proxy$_O$, it first verifies MTA$_O$ has subscribed to Call Transfer service. If so, it decrypts the Dcs-State information in the Dcs-Also header to determine the local gate location and identification. CMS/Proxy$_O$ queries the gate to obtain the transferred call's original billing information. CMS/Proxy$_O$ inserts billing information to indicate that the user associated with the number 212-555-1111 will pay for the new call segment. CMS/Proxy$_O$ extracts the call routing from the Dcs-state information, and then forwards the message to CMS/Proxy$_{T1}$.

(2) INVITE(also,replace):	*Description*
INVITE sip: Host(dp-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-State: Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	*State information as requested by CMS/Proxy$_{T2}$*
Dcs-Also: tel:+1-212-555-2222? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111 & Dcs-Billing-Info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> & Dcs-Billing-Info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333> & Dcs-Billing-ID= Host(dp-o.provider): 36123E5C:0152	*Original billing information will be used for a pseudo-call from originator to the point where the call was forwarded, and new billing information used for a pseudo-call from forwarding location to the new destination. Original billing identifier is kept for the forwarded call*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	*Call leg identification*
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	

CMS/Proxy$_{T2}$ forwards the INVITE(also,replace) message to MTA$_{T2}$ after encrypting the destination of the transfer, and the Dcs-Billing, Dcs-Billing-ID headers.

(3) INVITE(also,replace):	*Description*
INVITE sip: 555-3333@Host(mta-t2.provider) SIP/2.0	*Routing information obtained from Dcs-State header value*
Via: SIP/2.0/UDP Host(dp-t2.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*

Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Also: sip:{type=transfer; dest=tel:+1-212-555-2222; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> ; billing-info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>}$_K$@Host(dp-t2.provider);private ? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111	*Dcs-Also: contains the encrypted forwarder, new destination, Billing-identifier, timestamp, and Billing-Information fields. All are checksummed, signed by CMS/Proxy$_{T2}$, and encrypted.*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	*Call leg identification*
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	

MTA_{T2} acknowledges receipt and understanding of the INVITE(also,replace) by sending a 200-OK to MTA_O. This message is routed through the CMS/Proxy CMS/Proxy$_{T2}$, CMS/Proxy$_O$, and then delivered to MTA_O. MTA_O responds directly with an ACK. CMS/Proxy$_O$ is now done, while MTA_O is waiting for the BYE message, which will come after MTA_{T2} contacts the new destination.

(4) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t2.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers, as given in the INVITE message*
Dcs-State: Host(dp-t2.provider); state="{gate= Host(cmts-t2.provider): 4321/31S14621, nexthop=sip:Host(dp-o.provider), state="Host(dp-o.provider); nexthop=sip:555-1111@Host(dp-o.provider); gate=Host(cmts-o.provider):3612/3S10782; orig-dest=tel:+1-212-555-3333; num-redirects=0"}$_K$"	*State information stored at MTA$_{T2}$ regarding this call*
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	

CMS/Proxy$_{T2}$ restores the encrypted Via headers, and forwards the OK to topmost Via – CMS/Proxy$_O$.

(5) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	

CMS/Proxy$_O$ forwards the 200-OK to MTA_O.

(6) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	

MTA_O responds with an ACK message.

(7) ACK:	***Description***

ACK sip:Host(mta-t2.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	

After processing the INVITE(also,replace), MTA_{T2} issues a INVITE to MTA_{T1}. In addition to the standard headers carried in an INVITE message, the encrypted {Dcs-Billing, Dcs-Billing-ID} fields received in the INVITE(also,replace) message are copied into the Request-URI of the INVITE message. These fields indicate the destination, and that the user associated with the 212-555-1111 number will be charged for the second call leg.

(8) INVITE:	***Description***
INVITE sip:{type=transfer; dest=tel:+1-212-555-2222; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> ; billing-info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>}$_K$@Host(dp-t2.provider);private SIP/2.0	*Destination for the INVITE is taken from the Dcs-Also: header in the INVITE-REPLACE above. Private-param indicates the information is encrypted, and the first encrypted item, transfer, indicates the format.*
Via: SIP/2.0/UDP Host(mta-t2.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Smith <tel:555-3333>	*Originator supplied calling name and number*
Dcs-Anonymity: Off	
From: "Alien Blaster" <sip:B64(SHA-1(555-3333; time=36124172; seq=74))@localhost>	*Call leg identification information. Note that MTA_O does not know the identity of the new destination, and therefore uses a random To: header. Sequence number in SHA-1 hash incremented. Call-ID is from the Dcs-Also header*
To: sip:B64(SHA-1(555-3333; time=36124172; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124033;seq=72)@localhost	
Cseq: 129 INVITE	
Dcs-Replaces: tel:555-1111	
Contact: sip:Host(mta-t2.provider)	*Local address for further end-to-end signaling exchanges*
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t2.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

When the $CMS/Proxy_{T2}$ receives the INVITE it first decrypts the header information to find the real destination for the call. CMS/Proxy compares the current time against the timestamp in the encrypted string; if the request is too old, it is refused. It invokes the call routing logic to determine which CMS/Proxy ($CMS/Proxy_{T1}$) to which the INVITE needs to be routed. It also embeds two Dcs-Billing-Info headers in this message. The first one identifies the user associated with the E.164 number 212-555-2222 as paying for the initial call leg (212-555-2222/212-555-1111). This information was derived from the customer account information for the caller during the first call attempt. The second Dcs-Billing-Info header identifies the user associated with the E.164 number 212-555-1111 as paying for the second call leg (212-555-1111/212-555-3333), and was provided by $CMS/Proxy_O$ in the INVITE(also,replace) message.

(9) INVITE:	***Description***

INVITE sip: +1-212-555-2222,lrn=212-265@Host(dp-t1);user=np-queried SIP/2.0	*"lnp" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(dp-t2.provider); branch=1;	
Via: SIP/2.0/UDP Host(mta-t2.provider);	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-3333>	*Verified Caller Identification, with full E.164 number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-t2.provider):3612/17S30124/37FA1948	
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>	*IP address and encryption key of the record keeping server for event collection: account number/originating number/terminating number for billing. From the URI in INVITE message*
Dcs-Billing-Info: Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>	*Further billing information regarding split charging. Information from the URI in INVITE message.*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-3333; time=36124172; seq=74))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-3333; time=36124172; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124033;seq=72)@localhost	
Cseq: 129 INVITE	
Dcs-Replaces: tel:555-1111	
Contact: sip:Host(mta-t2.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_{T1}$ queries the directory server to determine the IP address (MTA$_{T1}$) associated with 212-555-2222. It then forwards the INVITE message to MTA$_{T1}$, after stripping off all of the billing fields, and adding the encrypted state information.

MTA$_{T1}$ recognizes the Call-ID matching an existing call, and matches the value of the Dcs-Replaces: header to the From/To of that call. Since they match, the call is allowed to proceed, with the 183-Session-Progress, receiving PRACK, PRECONDITION-MET, etc. These messages are identical to the basic call flow shown in Figure 29, and are not repeated here.

Upon sending of the 200-OK message, MTA$_{T1}$ processes the Dcs-Replaces header in the INVITE, and sends the following BYE message to the original destination.

(22) BYE:	***Description***
BYE sip:Host(mta-o.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-t1.provider)	
From: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	*Call leg identification of original call MTA$_{T1}$ to MTA$_O$*
To: tel:555-1111	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Cseq: 129 BYE	

Upon receipt of the BYE message, MTA$_T$ releases all network resources that have been used for this call. MTA$_T$ sends the following 200-OK message to MTA$_O$.

(23) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t1.provider)	
From: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	*Call leg identification*
To: tel:555-1111	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Cseq: 129 BYE	

Upon receiving of the 200-OK message, MTA_{T2} processes the Dcs-Replaces header in the INVITE, and sends the following BYE message to MTA_O.

(24) BYE:	***Description***
BYE sip:Host(mta-o.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-t2.provider)	
From: tel:555-3333	*Call leg identification of original call MTA_{O1} to MTA_{T2}*
To: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Cseq: 129 BYE	

Upon receipt of the BYE message, MTA_T releases all network resources that have been used for this call. MTA_T sends the following 200-OK message to MTA_O.

(25) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t2.provider)	
From: tel:555-3333	*Call leg identification*
To: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Cseq: 129 BYE	

Appendix P Three Way Calling With Network Bridge

Three-way calling is a fairly complex consumer service that allows a subscriber to simultaneously talk to two parties, and for those two parties to hear each other. It is often thought of as an ad-hoc conference bridge. Usage of the service proceeds as follows. The customer has an active call, either one initiated or received. The customer then does a hookflash, which places the existing call on hold and presents a dialtone. The user then dials the a second number, and connects to that party. A hookflash at this point creates a 3-way call, bridging the two calls together. Note the distinction between three-way calling and call waiting (where the two calls are alternately placed on hold and connected) lies in the fact that the subscriber initiated the second call; if the second call was an incoming call then the call-waiting service would be active.

The desired state during the three-way-call is three separate call legs, from each participant to the bridge server. If the participants initiate the calls, then they all have the same Call-ID, which tells the bridge to mix them together. If the bridge initiates the connections, there is no necessity for a common Call-ID. Multiple methods exist using combinations of Dcs-Also and Dcs-Replaces headers to achieve the desired connections. One way involves the subscriber establishing a connection to a bridge element, then transferring both of the existing calls to the bridge. Another method involves the subscriber asking the bridge to handle redirecting the existing calls to itself. The latter involves fewer signaling messages, and is preferred over the former. There is, of course, a third option – that the conference bridging function is done within the MTA and the network sees it as two separate simultaneous calls. As this consumes double the access network bandwidth, it is discouraged.

Initially a single call is active. For purposes of this example, consider that call to have been a call initiated by MTA_{T1} to MTA_O. The call identification information at MTA_O is as follows:

MTA_O state for call from MTA_{T1} to MTA_O	***Description***
From: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	Call leg identification
To: tel:555-1111	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Contact: sip: Host(mta-t1.provider)	Contact address for end-to-end signaling messages
Dcs-Remote-Party-ID: tel:+1-212-555-2222	Identity of other party in this call
Dcs-State: Host(dp-o.provider); state="{nexthop=sip:Host(dp-t.provider); gate=Host(cmts-o.provider):3612/17S30124; state="Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	State information of CMS/Proxy, encrypted and stored in MTA
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>	Billing information stored at gate
Dcs-Billing-ID: Host(dp-t1.provider):36124033:0381	Unique identifier for this call

MTA_O observes a hookflash and places this call on hold, issues a dialtone, and collects digits for a second call (212-555-3333). This sequence is shown in Figure 42, in Appendix O resulting in the first call being held and a conversation active to the second destination.

For this example, assume the second call is identified as follows:

State at MTA_O for call from MTA_O to MTA_{T2}	***Description***
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	Call leg identification
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
Contact: sip: Host(mta-t2.provider)	Contact address for end-to-end signaling messages
Dcs-Remote-Party-ID: tel:+1-212-555-3333	Identity of other party in this call

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/3S10782, nexthop=sip:+1-212-555-3333;lrn=212-256@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:555-3333@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	State information of CMS/Proxy, encrypted and stored in MTA
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>	Billing information stored at gate
Dcs-Billing-ID: Host(dp-o.provider):3612E5C:0152	Unique identifier for this call

The three-way-calling method described in this appendix asks the bridge to redirect the existing calls via an INVITE(Also). The bridge therefore is in control of managing the endpoints, and knows the proper media streams for mixing, even though they don't have a common Call-ID.

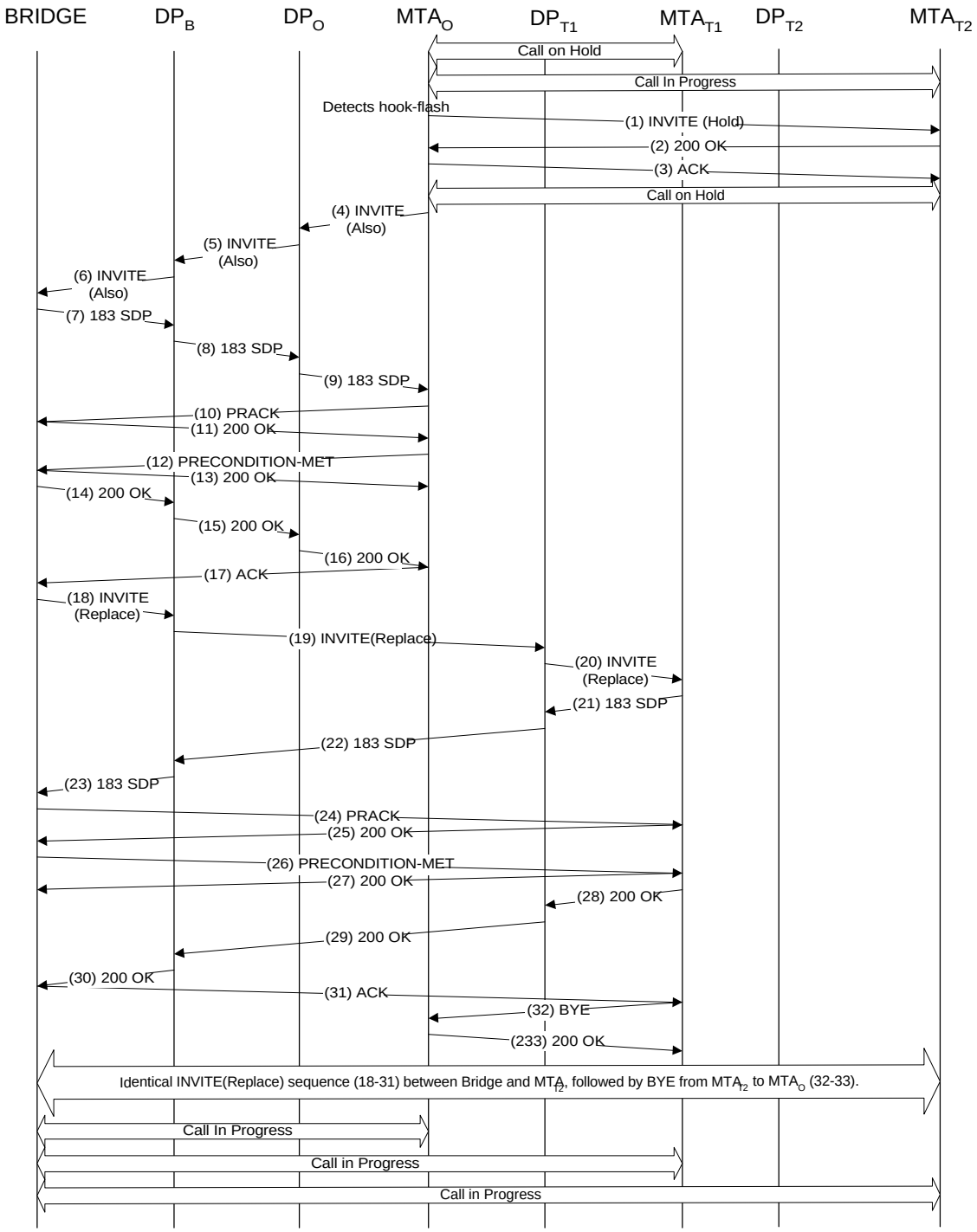


Figure 44: Three-Way-Call Signaling

Messages (1) to (3), for putting an existing call on hold, are identical to those used in Call Waiting (see Figure 40).

In response to the hook-flash, MTA also issues an INVITE to a bridge with a new call ID. The identity of the destination is given via the service name "bridge," which is a pre-defined service name in DCS.

(4) INVITE (Also):	***Description***
INVITE sip: bridge@Host(dp-o) SIP/2.0	*Request-URI uses keyword "bridge" as username to CMS/Proxy.*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
Dcs-Anonymity: Off	
Dcs-Also: tel:+1-212-555-2222 ? Call-ID=B64(SHA-1(555-2222;time=36124033;seq=72))@localhost & Dcs-Replaces=tel:555-1111 & Dcs-State= Host(dp-o.provider); state="{nexthop=sip:Host(dp-t.provider); gate=Host(cmts-o.provider):3612/17S30124; state="Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*Remote endpoint identification for the first call (incoming call to MTA$_O$), the Call-ID assigned to that call, and the To: header. Attached are the Dcs-State headers that match this call leg.*
Dcs-Also: tel:+1-222-555-3333 ? Call-ID=B64(SHA-1(555-1111;time=36124125;seq=23))@localhost & Dcs-Replaces=B64(SHA-1(555-1111;time-36124125;seq=23))@localhost & Dcs-State= Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/3S10782, nexthop=sip:+1-212-555-3333;lrn=212-256@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:555-3333@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*Remote endpoint identificcation for the second (originated by MTA$_O$) call, the Call-ID assigned to that call, and the From: header. Attached are the Dcs-State headers that match this call leg.*
From: sip:B64(SHA-1(555-1111; time=36124135;seq=24))@localhost	
To: sip: bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-o.provider)	
Cseq: 131 INVITE	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3460 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

CMS/Proxy$_O$ resolves the "bridge service name" to an available bridge (mcu41@Host(dp-b.provider) in this example), and forwards the INVITE to the associated CMS/Proxy (CMS/Proxy$_B$). In general, bridges will be available locally at CMS/Proxy$_O$, but this example demonstrates the messages exchanged if the bridge is remote. In general, bridges will be network services and located within the trusted domain of the network. However, they may also be provided by others. This example call flow diagram shows the latter case, where the bridge is outside the trusted domain of the service provider.

If the bridge is a trusted network element, the Bridge (for signaling purposes) would be functionally equivalent to a CMS/Agent, and use the same message set as is used between CMSs. In Figure 44 this would appear as if the lines CMS/Proxy$_B$ and BRIDGE were merged together.

CMS/Proxy$_O$ decrypts the state header values attached to the Dcs-Also headers, extracts the billing information for each of the previous call legs, and expands this information into the Dcs-Billing-Info values

(5) INVITE (Also):	***Description***
INVITE sip:mcu41@Host(dp-b.provider) SIP/2.0	*CMS/Proxy$_O$ resolves generic service to a particular server, CMS/Proxy$_B$*

Via: SIP/2.0/UDP Host(dp-o.provider)	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/5S12045/9142E7A1	
Dcs-Billing-Info: Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41@Host(dp-b.provider)/bridge-3>	*Billing information for the new call from subscriber to bridge*
Dcs-Billing-ID: Host(dp-o.provider):36124135:92	*Billing ID for new call from subscriber to bridge*
Dcs-Also: tel:+1-212-555-2222 ? CallID= B64(SHA-1(555-2222;time=36124033;seq=72))@localhost & Dcs-Billing-Info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> & Dcs-Billing-Info= Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41@Host(dp-b.provider)> & Dcs-Billing-ID= Host(dp-t1.provider):36124033:0381 & Dcs-Replaces=tel:555-1111	*Call information for the first call (incoming call to MTAO). Also header changed to be normal SIP URL, if a private one was provided by MTA_O, Billing information and Billing ID obtained from information stored at Gate. Second entry of billing information indicates split charging for the call to bridge*
Dcs-Also: tel:+1-212-555-3333 ? CallID= B64(SHA-1(555-1111;time=36124125;seq=23))@localhost & Dcs-Billing-Info= Host(rks-o.provider)/<5123-4567-8900/212-555-1111/212-555-3333> & Dcs-Billing-Info= Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41@Host(dp-b.provider)> & Dcs-Billing-ID= Host(dp-o.provider):36123E5C:0152 & Dcs-Replaces:B64(SHA-1(555-1111;time-36124125;seq=23))@localhost	*Call information for the second (originated by MTAO) call. Also header changed to be normal SIP URL, if a private one was provided by MTA_O, Billing information and Billing ID obtained from information stored at Gate. Second entry of billing information indicates split charging for the call to bridge.*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	*State information needed by CMS/Proxy-o for further message exchanges*
From: sip:B64(SHA-1(555-1111; time=36124135;seq=24))@localhost	
To: sip: bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-o.provider)	
Cseq: 131 INVITE	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
A=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3460 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

$CMS/Proxy_B$ encrypts the various fields into two Dcs-Also: headers, caches the Via headers, and passes the message to the bridge.

(6) INVITE (Also):	***Description***
INVITE sip:mcu41.provider SIP/2.0	
Via: SIP/2.0/UDP Host(dp-b.provider), {via=Host(dp-o.provider); via=Host(mta-o.provider)}$_K$	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
Dcs-Gate: 27S6028	*Local gate identification for this connection*
Dcs-Also: sip:{type=transfer; dest=+1-212-555-2222; Billing-info=Host(dp-t1.provider):36124033:0381; <timestamp>; Billing-info=Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>; Billing-id=Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41.provider>}$_K$@dp-b.provider;private ? Call-ID= B64(SHA-1(555-2222;time=36124033;seq=72))@localhost & Dcs-Replaces=tel:555-1111	*State information for the first call (incoming call to MTAO). Dcs-Also: header changed to be encrypted string opaque to server. Contents include requestor, call destination (with routing information), Billing-ID, and Billing-Info for the new call.*

Dcs-Also: sip:{type=transfer; dest=+1-212-555-3333; billing-id=Host(dp-o.provider):36123E5C:0152; expires=<timestamp>; billing-info=Host(rks-o.provider)/<5123-4567-8900/212-555-1111/212-555-3333>; billing-info=Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41.provider>}$_K$@dp-b.provider;private ? Call-ID= B64(SHA-1(555-1111;time=36124125;seq=23))@localhost & Dcs-Replaces:B64(SHA-1(555-1111;time-36124125;seq=23))	*State information for the second (originated by MTAO) call. Dcs-Also: header changed to be encrypted string opaque to server. Contents include requestor, call destination (with routing information), Billing-ID, and Billing-Info for the new call.*
Dcs-State: Host(dp-b.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-b.provider): 3612/27S6028; via="Host(dp-o.provider);branch=1", via=Host(mta-o.provider), state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124"}$_K$"	
From: sip:B64(SHA-1(555-1111; time=36124135;seq=24))@localhost	
To: sip: bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-o.provider)	
Cseq: 131 INVITE	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
A=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3460 RTP/AVP 0	
a=X-pc-codecs:96	

Bridge completes the call from MTA_O in a manner very similar to the basic call flow of Figure 29. Since a bridge doesn't need to alert a human, it responds immediately with 200-OK when resources are known to be available. Messages (7) through (17) are not detailed in this section.

Bridge initiates two calls in parallel, one to each of the participants listed in the Dcs-Also: headers. The Request URI in the new INVITE message is the encrypted string received in the Dcs-Also: header, the To: header is a generic string such as "participant<n>" since the bridge has no knowledge of the identity of the participants, and the Call-ID is the value from the Dcs-Also header. Part of the message sequence for MTA_{T1} (messages (18) to (33)) is detailed here; messages (34) through (49) are identical and not shown in the figure.

(18) INVITE (Replace):	***Description***
INVITE sip:{type=transfer; dest=+1-212-555-2222; Billing-info=Host(dp-t1.provider):36124033:0381; <timestamp>; Billing-info=Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>; Billing-id=Host(rks-o.provider)<5123-4567-8900/212-555-1111/mcu41.provider>}$_K$@dp-b.provider;private SIP/2.0	*Request URI is copied from one of the Dcs-Also: URLs, which is encrypted information from the CMS/Proxy that identifies the call to be made and the billing information for that call.*
Via: SIP/2.0/UDP Host(mcu41.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: Bridge Service <sip:Host(mcu41.provider)>	
Dcs-Anonymity: URL, Name	*Bridge requests that its URL and name not be given to the endpoints*
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	*From is a cryptographically random string chosen by the bridge. Call-ID is the value received in Dcs-Also. To is a generic string.*
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Contact: sip:Host(mcu41.provider)	
Cseq: 128 INVITE	
Dcs-Replaces:tel:555-1111	*Identity of Call Leg to be deleted, obtained from Dcs-Also header.*
Content-Type: application/sdp	
Content-length: (…)	

v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mcu41.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3174 RTP/AVP 0	
a=X-pc-Qos:mandatory sendrecv	
a=X-pc-codecs:96	

CMS/Proxy$_B$ decodes the encrypted Request-URI to find the real destination for this call. CMS/Proxy compares the current time against the expiration time in the encrypted string; if the request is too old it is refused. The call routing is determined from the destination contained in the encrypted string, as is the billing information for the call. CMS/Proxy$_B$ sends the following INVITE message to CMS/Proxy$_{T1}$:

(19) INVITE (Replace):	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(dp-t1.provider);user=np-queried SIP/2.0	*Request URI is obtained from the routing information*
Via: SIP/2.0/UDP Host(dp-b.provider)	
Via: SIP/2.0/UDP Host(mcu41.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: Bridge Service <sip:Host(mcu41.provider)>	*Caller-ID information verified by CMS/Proxy*
Dcs-Anonymity: URL, Name	
Dcs-Gate: Host(cmts-b.provider):3612/28S6029/079317A3	*Gate identification at Bridge end of connection*
Dcs-State: Host(dp-b.provider); nexthop=Host(mcu41.provider); gate=Host(cmts-b)::3621/28S6029; orig-dest=tel:+1-212-555-2222; num-redirects=0	*New state information generated by CMS/Proxy-b for this call*
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	*Call identification copied from request.*
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Contact: sip:Host(mcu41.provider)	
Cseq: 128 INVITE	
Dcs-Replaces:tel:555-1111	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(bridge.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
a=X-pc-qos:mandatory sendrecv	
m=audio 3174 RTP/AVP 0	

CMS/Proxy$_{T1}$ processes this exactly as a normal INVITE message, and passes the message to MTA$_{T1}$.

(20) INVITE (Replace):	***Description***
INVITE sip:555-2222@Host(mta-t1.provider) SIP/2.0	*Request URI is obtained from the translation information*
Via: SIP/2.0/UDP Host(dp-t1.provider), {via=Host(dp-b.provider); via=Host(mcu41.provider)}$_K$	*Via line added by CMS/Proxy$_{T1}$. Second and later Via lines encrypted by CMS/Proxy$_{T1}$*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: <sip:{type=rem-id; dest=sip:Host(mcu41.provider)}$_K$@Host(dp-t1.provider);private>	*Changed to a private URL due to privacy request of call originator*

Dcs-Media-Authorization: 5S32740	*Gate identification at local end of connection*
Dcs-State: Host(dp-t1.provider); state={nexthop=sip:Host(dp-b); gate=Host(cmts-t1:3621/53S32740;state="Host(dp-b.provider); nexthop=Host(mcu41.provider); gate=Host(cmts-b)::3621/28S6029; orig-dest=tel:+1-212-555-2222; num-redirects=0"}K	
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	*Call identification copied from request.*
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Contact: sip:Host(mcu41.provider)	
Cseq: 128 INVITE	
Dcs-Replaces:tel:555-1111	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mcu41.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3174 RTP/AVP 0	
a=X-pc-Qos:mandatory sendrecv	
a=X-pc-codecs:96	

MTA_{T1} notes the Call-ID: header, and determines that it has a call with that ID and that the Dcs-Replaces: header matches either the From: or To: value for the call. This INVITE is therefore interpreted as an update to that existing call. The provisional response (183-Session-Progress) is sent (21) to the local CMS/Proxy, who restores the encrypted Via: headers and sends it (22) to the originating CMS/Proxy, who passes it (23) to the bridge. These messages are identical to those of the basic call flow. The bridge responds with the PRACK message (24), as in the basic call flow. The bridge then performs the resource allocation and continues as in the basic call flow.

Upon receipt of the ACK message, MTA_{T1} sends a BYE message to its original caller. Note that this message has the From: and To: headers reversed from the incoming INVITE originally received for this call. If MTA_{T1} had initiated the call to MTA_O, then the From: and To: would match those in the initial INVITE.

(32) BYE:	***Description***
BYE sip:Host(mta-o.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-t1.provider)	
From: tel:555-1111	
To: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Cseq: 129 BYE	

Upon receipt of the BYE message, MTA_O sends the following 200-OK message to MTA_{T1}.

(33) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-t1.provider)	
From: tel:555-1111	
To: sip:B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Call-ID: B64(SHA-1(555-2222;time=36124033;seq=72))@localhost	
Cseq: 129 BYE	

The sequence of messages (34)-(49) is identical, and performs the same functions for the other leg of the three-way conference.

Appendix Q Three-Way Calling Hangup Sequences

There are two distinct hangup sequences that need to be detailed: hangup of a participant and hangup of the originator. The first results in a basic call between the originator and the remaining participant. The latter results in a hangup of all participants. Figure 45 shows the sequence for hangup of a participant, while Figure 46 shows the sequence after hangup of the originator.

For both of the following detail call flows, consider the initial state information to be the following:

MTA_O: Call from MTA_O to BRIDGE	**Description**
From: sip:B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	Call leg identification
To: sip:bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mcu41.provider)	Contact information for signaling messages to Bridge
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/12S52127, nexthop=sip: mcu41@Host(dp-b), state="Host(dp-b.provider); nexthop=sip:Host(mcu41.provider); gate=Host(cmts-b.provider):4321/31S14621; orig-dest=sip:mcu41@Host(dp-b); num-redirects=0"}$_K$"	State information for this call kept at MTA_O
Dcs-Billing-Info: Host(rks-o.provider)/341FE8B<5123-0123-4567-8900/212-555-1111/mcu41.provider/Bridge-3	Billing information located at Gate.

MTA_{T1}: Call from BRIDGE to MTA_{T1}	**Description**
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	Call leg identification
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24)) @localhost	
Contact: sip:Host(mcu41.provider)	Contact information for signaling messages to Bridge
Dcs-State: Host(dp-t1.provider); state="{nexthop=sip:Host(dp-b.provider); gate=Host(cmts-t1.provider): 3612/12S52127; state="Host(dp-b.provider); nexthop=sip:Host(mcu41.provider); gate=Host(cmts-b.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	State information for this call kept at MTA_{T1}
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>;Host(rks-o.provider)<5123-4567-8900/212-555-2222/mcu41.provider>	Billing information located at Gate.

MTA_{T2}: Call from BRIDGE to MTA_{T2}	**Description**
From: sip:B64(SHA-1(bridge;time=36124135;seq=312)) @localhost	Call leg identification
To: sip: participant2@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24)) @localhost	
Contact: sip:Host(mcu41.provider)	Contact information for signaling messages to Bridge
Dcs-State: Host(dp-t2.provider); state="{nexthop=sip:Host(dp-b.provider); gate=Host(cmts-t2.provider): 3612/13S52196; state="Host(dp-b.provider); nexthop=sip:Host(mcu41.provider); gate=Host(cmts-b.provider):3612/18S37224; orig-dest=tel:+1=212-555-3333; num-redirects=0"}$_K$"	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>;Host(rks-o.provider)<5123-4567-8900/212-555-3333/mcu41.provider>	Billing information located at Gate.

State information at the Bridge is:

Call from MTA_O to BRIDGE	**Description**
From: sip:B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	Call leg identification
To: sip:bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-o.provider)	Contact information for signaling messages to Bridge

Dcs-Remote-Party-ID: tel:+1-212-555-1111	
Dcs-State: Host(dp-b.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-b.provider):3612/15S30179; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	State information for this call kept at Bridge
Dcs-Billing-Info: Host(rks-o.provider)/341FE8B<5123-0123-4567-8900/212-555-1111/mcu41.provider/Bridge-3	Billing information located at Gate.
Call from BRIDGE to MTA$_{T1}$	**Description**
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	Call leg identification
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-t1.provider)	Contact information for signaling messages to Bridge
Dcs-State: Host(dp-b.provider); state="{gate= Host(cmts-b.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(DP-t1), state="Host(dp-t1.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t1.provider):4321/31S14621; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	State information for this call kept at Bridge$_1$
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>;Host(rks-o.provider)<5123-4567-8900/212-555-2222/mcu41.provider>	Billing information located at Gate.
Call from BRIDGE to MTA$_{T2}$	**Description**
From: sip:B64(SHA-1(bridge;time=36124135;seq=312))@localhost	Call leg identification
To: sip: participant2@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Contact: sip:Host(mta-t2.provider)	Contact information for signaling messages to Bridge
Dcs-State: Host(dp-b.provider); state="{gate= Host(cmts-b.provider): 3612/18S37624, nexthop=sip:+1-212-555-3333;lrn=212-234@Host(DP-t2), state="Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):3621/13S52196; orig-dest=tel:+1-212-555-3333; num-redirects=0"}$_K$"	State information for this call kept at Bridge
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>;Host(rks-o.provider)<5123-4567-8900/212-555-3333/mcu41.provider>	Billing information located at Gate.

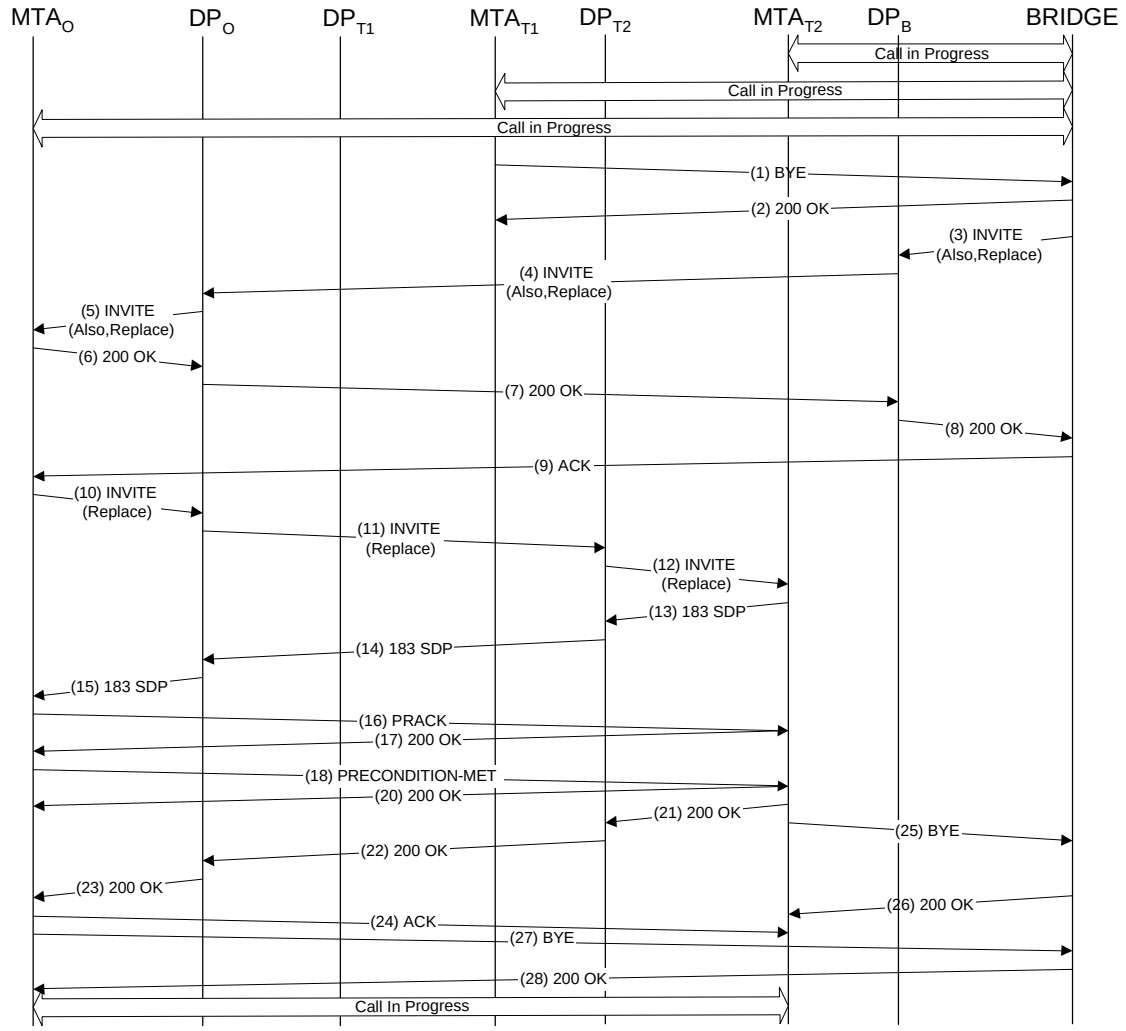


Figure 45: Three-Way-Call Signaling – Hangup of Participant

For this example, consider MTA_{T1} to drop out of the three-way conference. MTA_{T1} sends a BYE message to terminate its current call.

(1) BYE	***Description***
SIP/2.0 BYE Host(mcu41.provider)	*Request URI is taken from Contact: header*
Via: SIP/2..0/UDP Host(mta-t1.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	*Call Identification*
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24)) @localhost	
CSeq: 12002 BYE	

The Bridge responds to MTA_{T1} with the expected acknowledgement message.

(2) 200-OK	***Description***
SIP/2.0 200 OK	
Via: SIP/2..0/UDP Host(mta-t1.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	
To: sip: participant1@localhost	

Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24)) @localhost	
CSeq: 12002 BYE	

The Bridge now reconnects the two remaining parties into a simple call. Since MTA_O is the originator of the three-way-call, the Bridge informs it of the need to redirect the call from MTA_{T2}. Bridge sends the INVITE(Also,Replace) message, via $CMS/Proxy_B$.

(3) INVITE(also,replace)	***Description***
INVITE sip:Host(mta-o.provider) SIP/2.0	*Request URI contains the Contact header of the call being changed*
Via: SIP/2.0/UDP Host(mcu41.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Also: tel:+1-212-555-3333 ? Call-ID= B64(SHA-1(555-1111;time=36124135;seq=24))@localhost & Dcs-Replaces= sip:B64(SHA-1(bridge;time=36124135;seq=312))@localhost & Dcs-State= Host(dp-b.provider); state="{gate= Host(cmts-b.provider): 3612/18S37624, nexthop=sip:+1-212-555-3333;lrn=212-234@Host(DP-t2), state="Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):3621/13S52196; orig-dest=tel:+1-212-555-3333; num-redirects=0"}$_K$"	*Also header contains the URL of the endpoint MTA_O is to contact. Call-ID is the ID of the call from Bridge to MTA_{T2}, Replaces is the From/To of the bridge of the call from Bridge to MTA_{T2}. Dcs-State headers matching that call leg are added.*
Dcs-State: Host(dp-b.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-b.provider):3612/15S30179; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State information for this call kept at Bridge*
From: sip:bridge@Host(dp-o.provider)	
To: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 12301 INVITE	
Dcs-Replaces: sip:bridge@Host(dp-o.provider)	*Identifies existing call leg to be torn down*

The call flow from this point onward is identical to the Call Transfer with Consultation, as shown in Figure 43. The bridge, having "consulted" with MTA_O, transfers its call with MTA_{T2} to MTA_O.

When the originator of a three-way call hangs up, the entire call is terminated. The bridge recognizes the BYE from the originator and sends BYE messages to all participants.

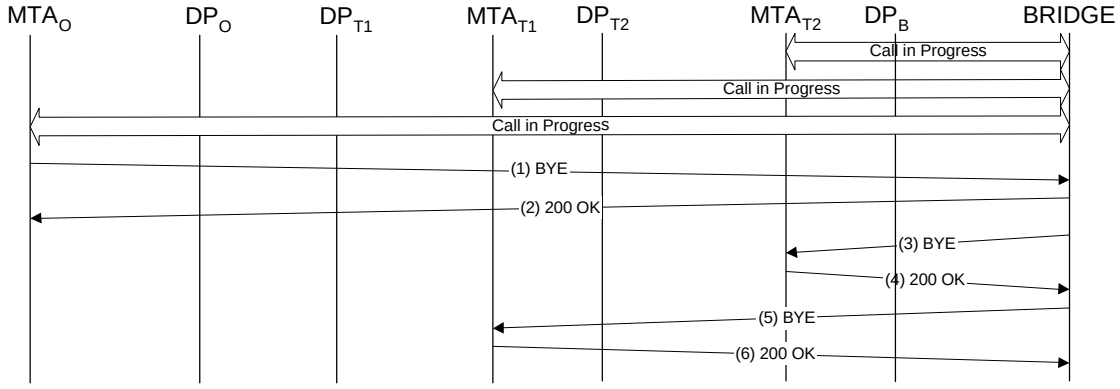


Figure 46: Three-Way-Call Signaling – Hangup of Originator

MTA_O -> Bridge

(1) BYE	***Description***
BYE sip:Host(mcu41.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: sip:B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
To: sip:bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 141 BYE	

Bridge -> MTA_O

(2) 200-OK	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: sip:B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
To: sip:bridge@Host(dp-o.provider)	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
Cseq: 141 BYE	

Bridge -> MTA_{T2}

(3) BYE	***Description***
BYE sip:Host(mta-t2.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mcu41.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=312))@localhost	
To: sip: participant2@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 80001 BYE	

MTA_{T2} -> Bridge

(4) 200-OK	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mcu41.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=312))@localhost	
To: sip: participant2@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 80001 BYE	

Bridge -> MTA_{T1}

(5) BYE	***Description***
BYE sip:Host(mta-t1.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mcu41.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 80002 BYE	

MTA_{T1} -> Bridge

(6) 200-OK	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mcu41.provider)	
From: sip:B64(SHA-1(bridge;time=36124135;seq=311))@localhost	
To: sip: participant1@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124135;seq=24))@localhost	
CSeq: 80002 BYE	

Appendix R CODEC Change within previous autho r ization

When the initial INVITE SDP contained multiple CODECs, such that any single CODEC would be authorized, no further interaction is needed with the CMS/Proxies to change CODECs. However, due to the requirements of the segmented reservation model of D-QoS, it is necessary to signal to the far end and synchronize changes in CODEC usage. Figure 47 shows the sequence of signaling messages to perform this function.

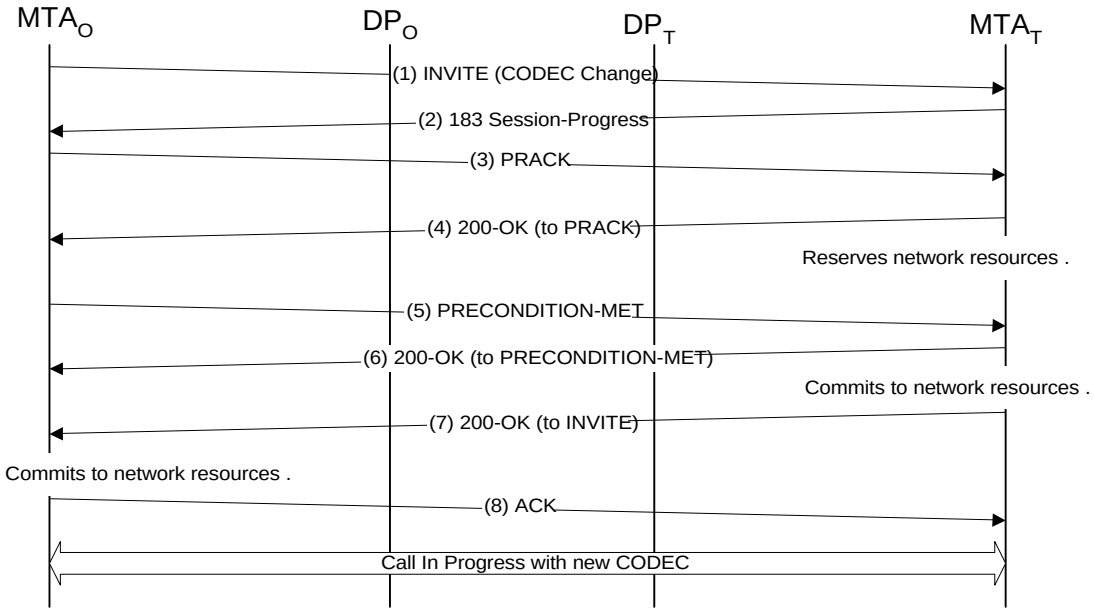


Figure 47: CODEC Change within previous Authorization

By some mechanism outside the scope of the Distributed Call Signaling protocol, MTA_O decides that a CODEC change is necessary. MTA_O sends the following INVITE message directly to MTA_T. This INVITE is almost identical to the initial INVITE that established the call, except for header fields such as Dcs-Remote-Party-ID and Dcs-Anonymity that are not sent to maintain originator privacy.

(1) INVITE:	***Description***
INVITE sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of Initial INVITE or initial 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost >	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	*CSeq is one higher than most recently issued INVITE request*
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (…)	
v=0	*Proposed new session description, with the new codec requested..*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	

a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	

Upon receipt of the (1) INVITE message, MTA_T verifies its ability to switch to the designated CODEC, and responds with a 183-Session-Progress including an updated SDP. The procedures from this point onward are identical to a basic call flow, as given in previous Appendicies.

The resource reservations done in this call flow maintain the previous resources, so that if either end fails to make the proper reservation, the original call can proceed with the initial CODEC.

MTA_T actually switches to the new codec upon sending the final 200-OK response to the INVITE, and MTA_O switches to the new codec upon receipt of the final 200-OK.

Appendix S CODEC Change requiring new author ization

When an MTA wishes to change to a different CODEC, but that CODEC was not among those initially authorized (or subsequently authorized by this sequence), it is necessary to request an increased authorization from the CMS/Proxy. Figure 48 shows the sequence of signaling messages that achieves this.

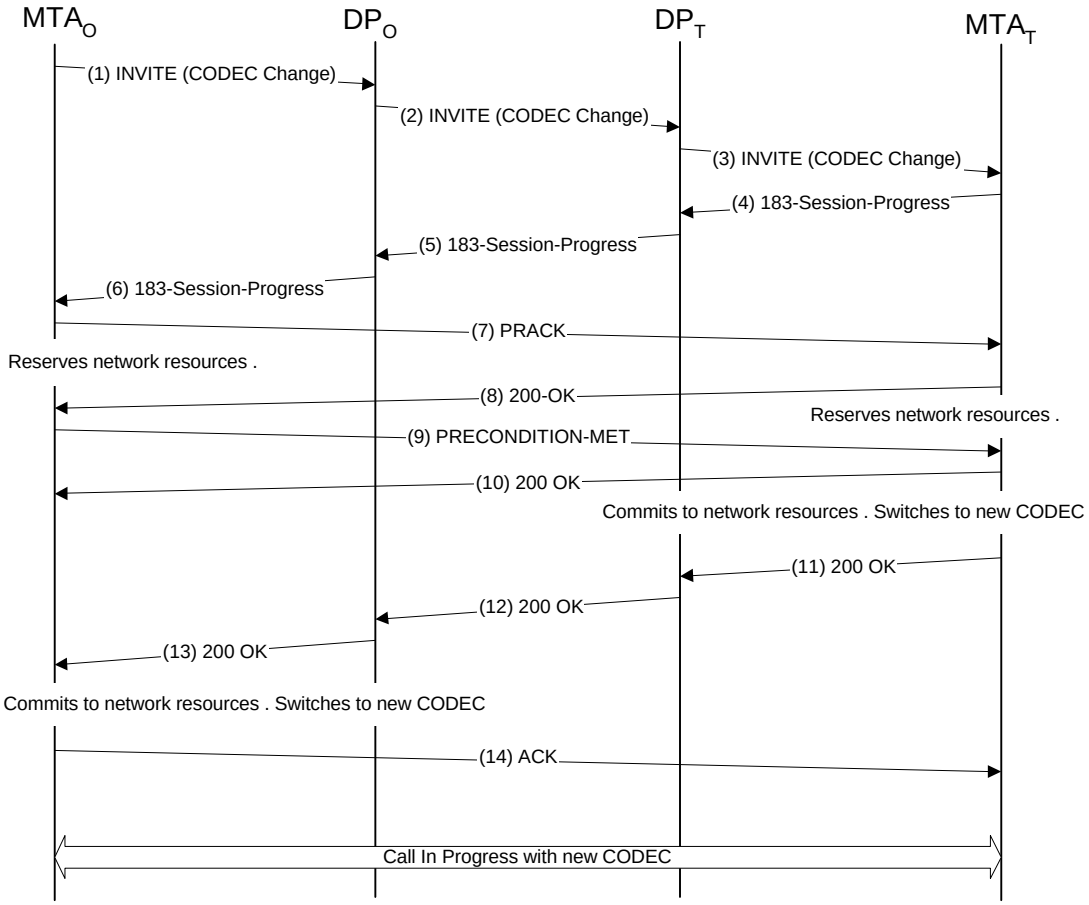


Figure 48: CODEC Change Requiring Authorization

By some mechanism outside the scope of the Distributed Call Signaling protocol, MTA_o decides that a CODEC change is necessary, and that the previous authorization for the current call does not permit this new CODEC (e.g. the initial call setup used only G.726-32 and the new CODEC desired is G.711). MTA_O generates the following SIP INVITE message and sends it to $CMS/Proxy_O$ (the CMS/Proxy that manages MTA_O). MTA_O starts timer (T-proxy-request).

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(DP-o);user=phone SIP/2.0	*Request URI same as initial INVITE.*
Via: SIP/2.0/UDP Host(mta-o.provider)	*IP Address or Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State information for this call, as given by CMS/Proxy$_O$*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call identification same as previous.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	*CSeq is one higher than most recently issued INVITE request*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line). Note this SDP already contains the cipher suites and key information.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv	

Upon receiving the INVITE message, CMS/Proxy$_O$ authenticates MTA$_O$ using standard IPSec authentication. CMS/Proxy$_O$ decodes the state string in the Dcs-State header and extracts the relevant information about the current call. CMS/Proxy$_O$ generates the following INVITE message and sends it to CMS/Proxy$_T$. CMS/Proxy$_O$ adds a number of parameters to the INVITE message. These are described below.

(2) INVITE:	***Description***
INVITE sip: +1-212-555-2222,lrn=212-234@Host(DP-t);user=np-queried SIP/2.0	*New Request URI indicates that CMS/Proxy-t is destination of this message.*
Via: SIP/2.0/UDP Host(DP-o.provider);branch=1	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Gate: Host(cmts-o.provider): 3612/17S30124	*IP addr of CMTS, ID of the terminating gate, and key for gate coord. This information obtained from CMTS$_O$*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing. Information obtained from CMTS$_O$.*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#. Information obtained from CMTS$_O$.*
Dcs-State: Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	*State information for other proxies on the path*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 INVITE	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	

b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv	

Upon receiving this INVITE message, CMS/Proxy$_T$ recognizes this as a mid-call change by the presence of the Dcs-State header with its name attached, and generates the following INVITE message and sends it to MTA$_T$. Note that the Via lines may be different from the initial INVITE exchange; they have been encrypted to maintain the privacy of the caller.

(3) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Request URI indicates mta$_T$ is destination of this message. Obtained from Dcs-State value*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider);branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Media-Authorization: 31S14621	*Gate ID*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 INVITE	
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of media stream to be received by MTA$_O$.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv	

Upon receiving this INVITE, MTA$_T$ authenticates that the message came from CMS/Proxy$_T$ using IPSec. MTA$_T$ checks for a current call matching the triple (From, To, Call-ID). MTA$_T$ looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA$_T$ generates the following 183-Session-Progress response, and sends it to CMS/Proxy$_T$.

(4) 183 Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider);branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as presented in INVITE message.*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State header that matches the call-leg identification.*
Dcs-Remote-Party-ID: John Smith <tel: 555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	

Cseq: 129 INVITE	
Session: qos	
Contact: sip:Host(mta-t.provider)	*Address for future direct signaling messages to MTA_T*
Content-Type: application/sdp	*The response to INVITE in SIP must contain the SDP description of the media stream to be sent to MTA_T.*
Content-length: (…)	
v=0	*SDP contains the MTA_T bearer channel IP address, and negotiated voice encoding parameters*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv confirm	

Upon receiving the 200-OK message, CMS/$Proxy_T$ authorizes the resources and forwards the following 183-Session-Progress to CMS/$Proxy_O$, restoring the Via headers. At this point CMS/$Proxy_T$ has completed its transaction and does not maintain any more state for this call. CMS/$Proxy_T$ may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(5) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information previously requested by dp-o*
Dcs-Remote-Party-ID: John Smith <tel: +1-212-555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 INVITE	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/$Proxy_O$ authorizes the resources and forwards the following message to MTA_O. At this point CMS/$Proxy_O$ has completed its transaction and does not maintain any more state for this call.

(6) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	

Dcs-Remote-Party-ID: John Smith <tel: +1-212-555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 INVITE	
Session: qos	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	*Default key value to use for encrypting the data stream*
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, MTA_O sends the PRACK message directly to MTA_T using the IP address in the Contact header. MTA_O then reserves the resources needed, and sends a PRECONDITION-MET message if successful. The PRECONDITION-MET message, and the 200-OK messages in response to the PRACK and PRECONDITION-MET are identical to that in the basic call flow (Figure 29) and not shown here.

(7) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 130 PRACK	*Message being acknowledged*
v=0	*SDP description of final negotiated media stream.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos: mandatory sendrecv	

MTA_T reserves the resources as needed from the final SDP from the PRACK message. If successful, and upon receipt of the PRECONDITION-MET message from MTA_O indicating it was successful, MTA_T changes the CODEC and sends the 200-OK final response.

(11) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider);branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as presented in INVITE message.*

Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State header that matches the call-leg identification.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	

Upon receiving the 200-OK message, CMS/Proxy$_T$ forwards the following 200-OK to CMS/Proxy$_O$, restoring the Via headers.

(12) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information previously requested by dp-o*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	

Upon receiving the 200-OK message, CMS/Proxy$_O$ forwards the following 200-OK to MTA$_O$.

(13) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: Sip/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 INVITE	

Upon receiving the 200-OK message, MTA$_O$ sends the following ACK message directly to MTA$_T$ using the IP address in the Contact header of the previous 183 message. This completes the three-way handshake for the SIP INVITE exchange.

(14) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 129 ACK	*Message being acknowledged*

Appendix T Some MTA Value-Added Feature Possibilities

This appendix describes a number of features that may be implemented locally in an MTA. Many of these features are currently implemented in the central switches of the PSTN; the DCS architecture allows them to be implemented locally and reduce the processing load at the proxy servers.

Distinctive Ringing

This feature utilizes the caller-id information received with an incoming call, examines the calling number, and selects a ringing pattern based on the caller's identity. The definition of calling numbers, and ringing patterns, are locally stored in the MTA.

Anonymous Call Reject

For a customer that subscribes to Caller-ID, an incoming call that requests privacy is marked with the reserved string "private". The customer may desire to reject all such calls without any alerting, or with a short "ping-ring." If this feature is active, the MTA checks the caller-id information and rejects all calls that do not supply their calling number information.

Answering Machine/Voicemail/Message Waiting Indicator

The MTA, with sufficient memory, may serve as a complete answering machine. Incoming calls that are not answered with a preset number of rings can be automatically answered, played a local greeting message, and recorded a message in local memory. Presence of messages can light a Message Waiting Indicator, and can cause a special dialtone signal (e.g. stutter dialtone) when the phone is offhook.

Local announcements

The MTA can serve as a message board for family members. By dialing a special code, a user may request a playout of any recorded messages left for that individual. By dialing another special code, the user may record an audio message for another specific individual, or generally for everyone.

Home Intercom

By dialing a special code, local connections can be established between lines attached to the MTA. This "call" may result in ringing all the phones that are not offhook, and connecting all together in a local conference bridge.

Home PBX

As an extension of Home Intercom, the MTA can implement a small PBX, with the ability to do abbreviated dialing for other extensions and to locally make the connections.

Multiple calls from single phone line

Multi-button keysets, dual-line phones, etc, may all be handled locally by the MTA. Selecting one of a number of buttons provides an extended call-waiting feature allowing more than a single call to be placed on hold.

Call queuing/parking/camp-on (without any e-e signaling)

Calls received while the phone is busy can be automatically answered by the MTA, and played a short "Call right back" message. The Dcs-State variable is saved for each. When the phone becomes idle, the MTA initiates a return-call for each, in turn. This has the effect of a queue of waiting calls, which are returned in the order received.

Alternatively, the MTA can automatically answer incoming calls, play a short message, and place them on hold. The MTA can periodically revisit each held call, giving a message indicating their position in the current queue. These held calls can be answered in the order received, or in a different order based on a priority determined from the caller-id.

Call blocking (900/976/long distance/…)

Various forms of call blocking, e.g. 976, 900, long-distance, international, can be implemented in the MTA, with a pre-recorded message indicating that the call is blocked. Due to the potential of rogue software in the MTA, this blocking function cannot be guaranteed, and may still need to be implemented in the proxy as well.

Speed-dialing

The MTA can store a large list of speed-dialing numbers, which can be accessed by dialing a short code.

Selective call forwarding

This feature utilizes the caller-ID information available to subscribers of that service. Based on locally configured information, the MTA is able to selectively forward calls to multiple different destinations, based on the identity of the caller. Such forwarding may also depend on time of day, or presence of specific people in the house.

Call waiting disable

Call waiting tone block (*70 in the current PSTN) can be implemented locally in the MTA. If this feature is active, the MTA can refuse incoming calls rather than give the normal call waiting alerting tone.

Hot-line

This feature causes a pre-defined number to be automatically dialed whenever the phone is lifted offhook. This is commonly used for emergency phones, but can be used in other situations as well.

Voice dialing

An intelligent MTA can listen for key words or phrases in speech, and react by automatically dialing preset numbers.

Appendix U E911 Call Flow

The signaling for a call to 911 is handled by the MTA identically to a basic call. At the CMS/Proxy the call will likely be given a higher priority. Routing of 911 calls, in the near term, will be to a PSTN Gateway, which will route them over a special MF trunk group. In determining the proper routing number/office code to send over the MF trunk, the CMS/Proxy must perform a translation based on the originating number, so the call completes at the proper emergency services office.

Two changes are implemented in the MTA in support of 911. First, call waiting is disabled so any incoming call to the MTA is given a BUSY error instead of call-waiting treatment. Second, if the user goes onhook, the MTA maintains the connection to the Emergency Service Provider until a BYE is received from them.

Since the MTA is under the customer control, there can be no assurances that the above service modifications will be performed. Therefore the service more closely follows the cellular model, where the customer retains ultimate control of the call, and not the LEC model where the 911 operator has control.

Appendix V Operator Busy Line Verification Call Flow

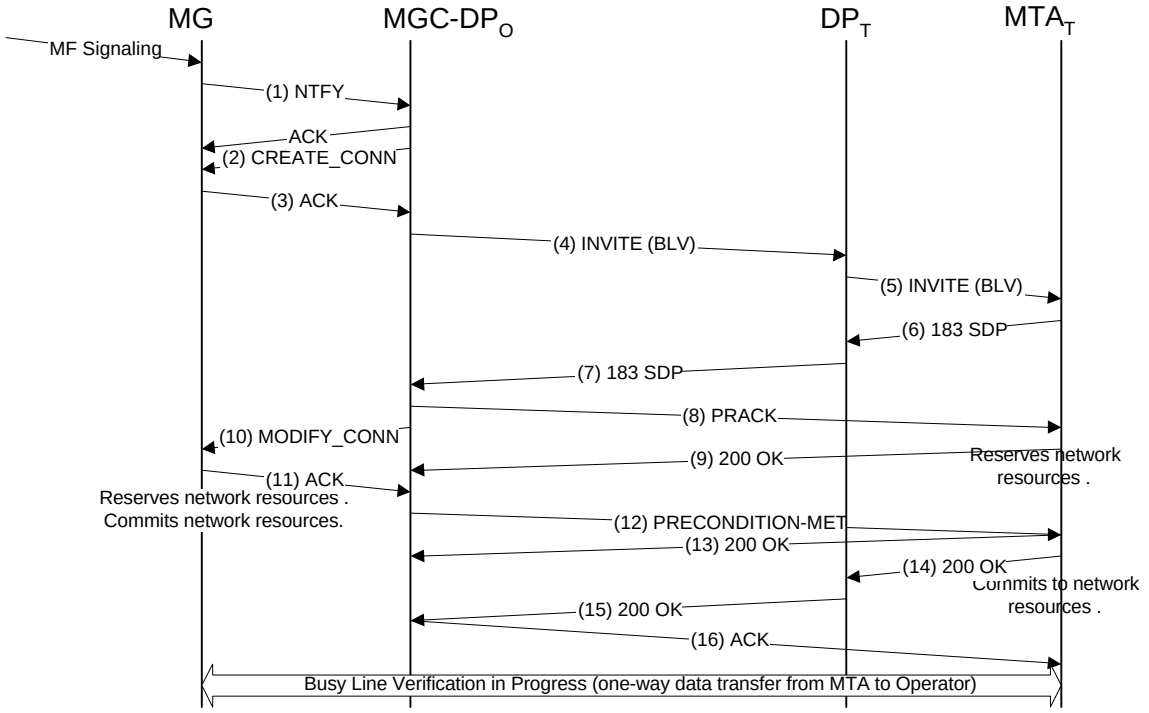


Figure 49: Busy Line Verification Call Flow

This service clearly requires the cooperation of the MTA. Further, there is a network database of phone numbers of customers who cannot be verified and/or broken into. It seems reasonable that a MTA that refuses to cooperate is merely one so marked in the current database.

Busy Line Verification sequence begins when the Operator at an OSPS console signals an E.164 number for verification over a special MF trunk group, to the PSTN gateway. The Media Gateway (MG) and Media Gateway Controller (MGC) recognize this special signaling, and generate an INVITE(BLV) to the number requested.

The normal call initiation sequence in TGCP is followed. The NTFY message (1) signals the MGC of a call request, and MGC uses the CRCX message (2) and ACK (3) to generate an appropriate SDP. That it is a OSPS trunk group is known to the MGC, which invokes the special header insertion.

MGC recognizes the trunk group as special BLV trunks, and generates a slightly modified INVITE message, by adding the Dcs-OSPS header.

(4) INVITE (BLV):	***Description***
INVITE sip:212-555-1111,lnp=212-237@dp-t.provider;user=phone SIP/2.0	*lnp-parm shows that LNP dip done and 6 digits is LRN*
Via: SIP/2.0/UDP Host(mgc02.provider);branch=1	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: Operator <sip:Operator42@mgc02.provider>; rpi-type=operator	*Calling Name, Caller ID, and Caller-Type (Operator)*

Dcs-Anonymity: URL	*URL of operator to be hidden*
Dcs-OSPS: BLV	*Indicator of Busy Line Verification function*
Dcs-Gate: mgc02.provider/36123E5B	*IP address of the originating gate (the originating CMTS)*
Dcs-Billing-Info: Host(rks-o.provider)<OSPS/212-0/212-555-1111>	*IP address and encryption key of the record keeping server for event collection: account number/originating number/terminating number for billing*
Dcs-Billing-ID: Host(mgc02.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: B64(SHA-1(0;time=36123E5B;seq=72))@localhost	*The triple (From, To, CallID) is used by SIP to uniquely identify a call*
To: tel:555-1111	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 127 INVITE	
Contact: sip:mgc02.provider	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 mg101.provider	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3380 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

CMS/Proxy$_T$ authorizes the additional connection without regard to number of currently active connections, and passes the INVITE(BLV) to MTA$_T$.

(5) INVITE (BLV):	***Description***
INVITE sip:212-555-1111@mta-t.provider;user=phone SIP/2.0	*lnp-parm shows that LNP dip done and 6 digits is LRN*
Via: SIP/2.0/UDP Host(dp-t.provider);branch=1, { via="Host(mgc02.provider);branch=1"}$_K$	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: Operator <sip:{type=remote-id; orig=sip:Operator42@mgc02.provider; anonymity=URL}$_K$>; rpi-type=operator	*Calling Name, Caller URL, and Caller-Type (Operator). The URL is encrypted, and contains the anonymity flag.*
Dcs-OSPS: BLV	*Indicator of Busy Line Verification function*
Dcs-Gate: 44S10312	*ID of the terminating gate (the terminating CMTS)*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(mgc02. Provider); gate=Host(cmts-t.provider):4321/44S10312}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, and all previous state headers from other proxies*
From: B64(SHA-1(0;time=36123E5B;seq=72))@localhost	*The triple (From, To, CallID) is used by SIP to uniquely identify a call*
To: tel:555-1111	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:mgc02.provider	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 mg101.provider	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	

m=audio 3380 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

MTA_T does not respond to a BLV request with BUSY, nor does it perform call forwarding. The response is 183-Session-Progress, identical to that of a normal call given in Appendix B, and completes identical to a normal call (but without the ringing phase).

MTA_T commits to the reserved resources, and begins to send voice packets to the Operator. The payload contains a copy of the packets generated at MTA_T.

If the designated line is onhook, MTA_T will generate silence packets and send to Operator. If the line is currently ringing or generating local ringback, MTA_T will generate a ringback tone pattern and sent to Operator.

The OSPS system scrambles the received voice packets, making the conversation unintelligible to the Operator. However, enough information is passed through the scrambled audio for the operator to accurately determine whether the line is in use, dialing, ringing, or idle.

A BLV call terminates when the OSPS signals a hangup over the MF trunk, resulting in a DCS BYE message. The MTA never terminates a BLV call.

Appendix W Operator Break-In Call Flow

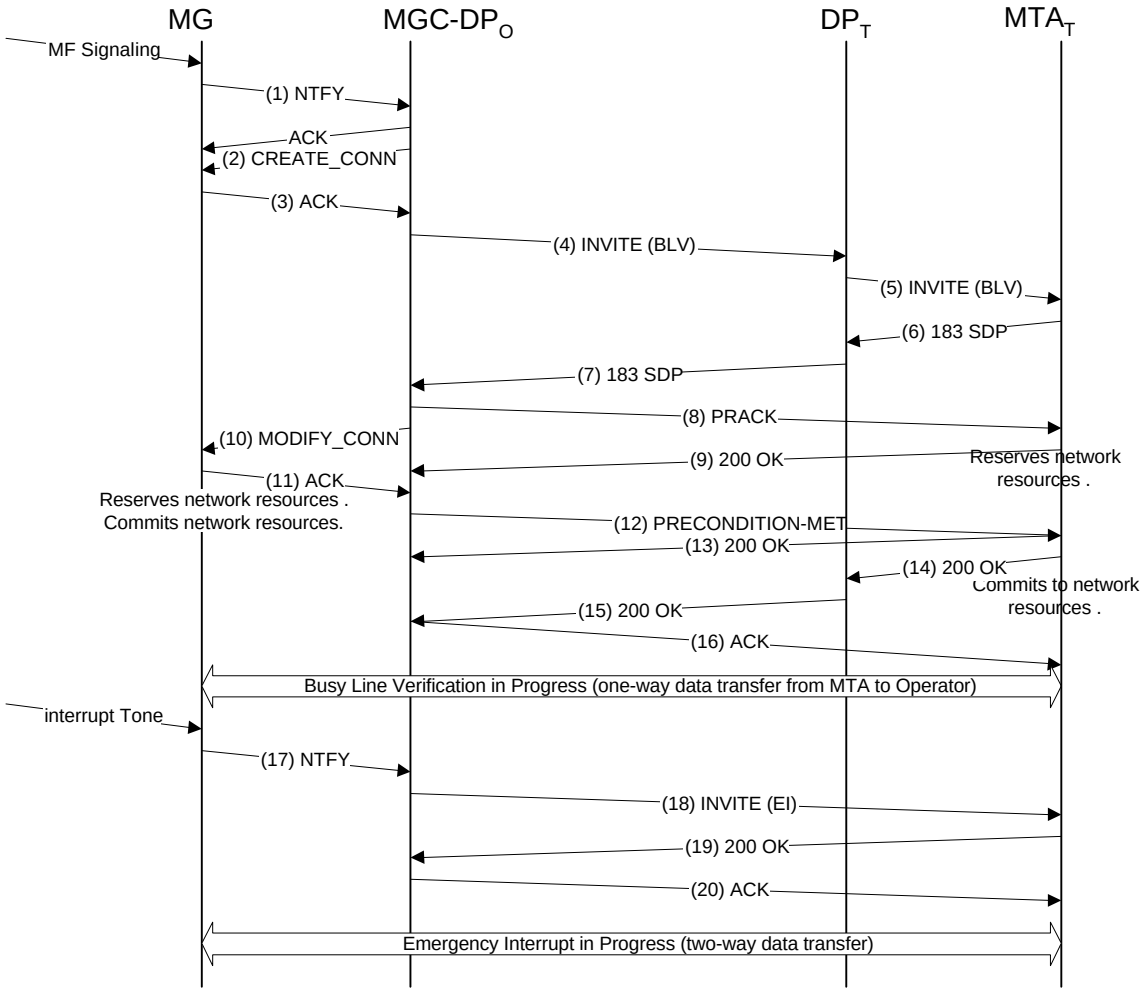


Figure 50: Emergency Interrupt Call Flow

Emergency Interrupt is closely tied to BLV (previous appendix), since EI always begins with BLV. Messages (1) through (16) perform the BLV function, and are not repeated here.

At the end of the BLV call flow, instead of the OSPS releasing the trunk, OSPS generates an alerting tone. The Media Gateway (MG) detects activity on line and the Media Gateway Controller (MGC) generates INVITE(EI) and sends it direct to MTA$_T$.

(18) INVITE(EI):	***Description***
INVITE sip:Host(mta-t.provider) SIP/2.0	*Request-URI is the Contact header*
Via: SIP/2.0/UDP Host(mgc02.provider)	
From: B64(SHA-1(0;time=36123E5B;seq=72))@localhost	*Call leg identification. These three fields must match those used in the initial INVITE(BLV) message.*
To: tel:555-1111	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 130 INVITE	*CSeq one greater than previous INVITE*

Dcs-OSPS: EI	*Indicates a change to Emergency Interrupt*
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow. This SDP replaces the previous one.*
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 mgc02.provider	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3380 RTP/AVP 0	

MTA verifies that BLV is already active, and that the EI request matches From, To, Call-ID. If so, it responds with 200-OK. SDP is not needed unless there is a change in the session description from that of the BLV.

(19) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mgc02.provider)	
From: B64(SHA-1(0;time=36123E5B;seq=72))@localhost	
To: tel:555-1111	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 130 INVITE	

MGC responds with the standard SIP ACK message.

MTA has several choices of how to do the EI function. 1) put current call on hold and connect user to operator. 2) perform local mixing of current call and stream from OSPS, so both participants in existing call hear and can talk to the operator. 3) allocate a bridge and treat this just like a 3-way call.

Appendix X Lawfully Authorized Electronic Surveillance Call Flow

Calls from and to surveillance subjects behave just like a normal call flows. Additional messages will be sent from the CMS to the Electronic Surveillance Delivery Function telling the known signaling information. One additional parameter is sent to the CMTS in the Gate-Set command, telling it to copy all the voice data packets and send the copy to the Law Enforcement Access Point.

There is no change to the MTA-CMS message exchanges due to electronic surveillance. This is an absolute requirement due to the non-intrusive requirement of the electronic surveillance statute.

In most cases, there is no change to the CMS-CMS message exchanges due to electronic surveillance. This basic design keeps the knowledge of surveillance as localized as possible. If a call originator is under surveillance, the surveillance is done at the originating CMTS; the call destination does not know in any way that it is happening. If a call destination is under surveillance, the surveillance is done at the terminating CMTS; the call originator does not know in any way that it is happening. If both the call originator and call destination are under surveillance, the interception is done twice. So it goes.

The only situation where CMS-CMS message exchanges are extended to support electronic surveillance is in cases of call redirection. When a subject under surveillance initiates a call transfer, it is required that the new call also be intercepted. Therefore the notification of surveillance is passed in that CMS-CMS INVITE message. Further, when the new destination is unable to perform the required interception (e.g. redirection to a network server such as voicemail), the 183 response contains additional information telling the originator to perform the surveillance. These situations are shown in the following examples.

Call Forwarding (Unconditional) with Forwarder under Surveillance

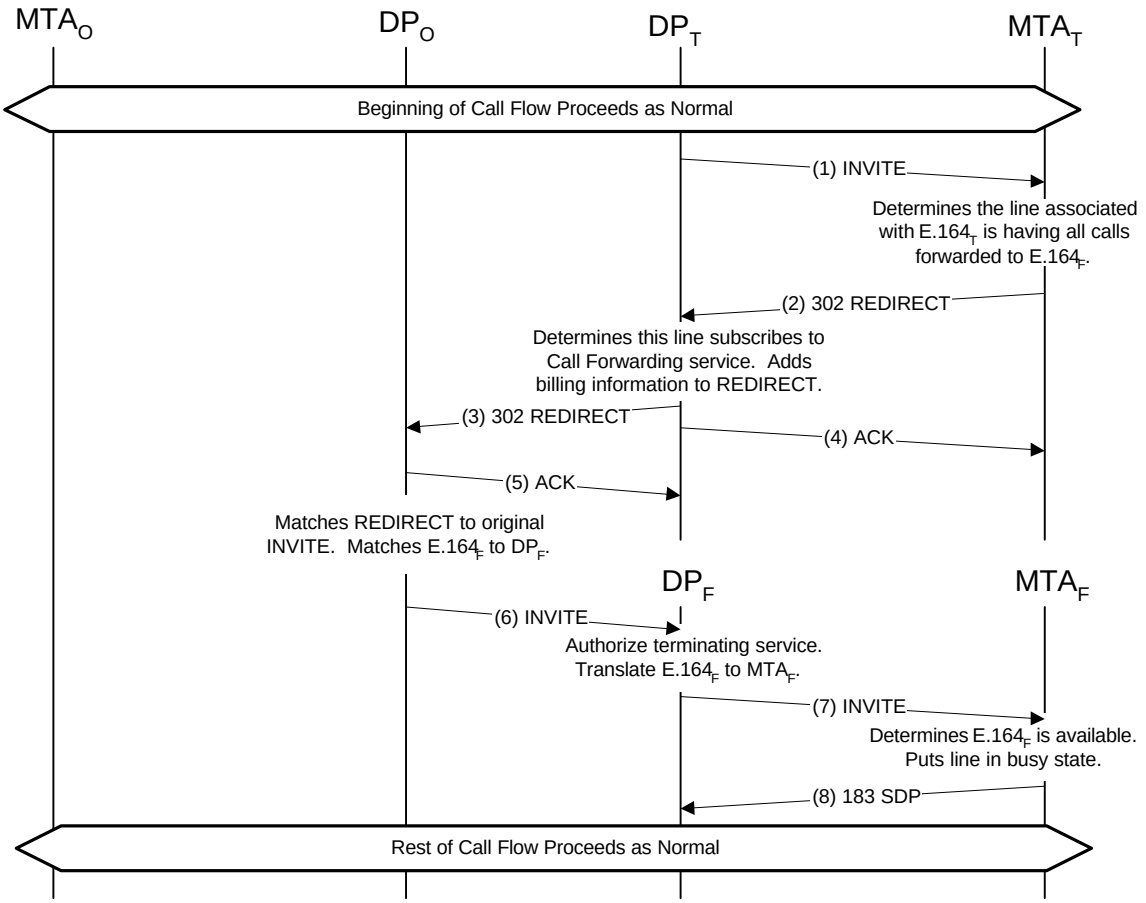


Figure 51: Call Forwarding Unconditional while under Surveillance

For this example, consider a call from MTA_O to MTA_T (with MTA_T under surveillance). MTA_T has established call forwarding-unconditional. The basic call flow for call-forwarding is identical to that given in Appendix F, and only the differences are noted here.

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
Dcs-Media-Authorization: 31S14621	

Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this message, MTA_T determines that the line associated with 212-555-2222 is having all calls forwarded. It issues a REDIRECT (302) response to indicate that it wants the call forwarded. This message carries the forwarding number in the Contact header.

(2) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	
Dcs-Anonymity: off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:555-3333	

$CMS/Proxy_T$ knows that MTA_T is under surveillance, and includes the Dcs-Laes header in the 302-Redirect response sent back to CMS_O. This header contains the delivery function information needed by the new destination of the call.

(3) 302-Redirect	***Description***
SIP/2.0 302 Moved Temporarily	
Via: SIP/2.0/UDP Host(dp-o.provider); branch = 1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	

Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
Dcs-Anonymity: off	
Dcs-Laes: Host(df-t)/Host(df-t);surveillancekey	*Surveillance information for MTA$_T$*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: tel:+1-212-555-3333	

CMS/Proxy$_O$ determines the CMS/Proxy$_F$ for the E.164 number 212-555-3333 when it receives the 302-Redirect message. It generates an INVITE message and sends it to CMS/Proxy$_F$. CMS/Proxy$_O$ adds the Dcs-Laes header to this INVITE, with the delivery function information received from CMS/Proxy$_T$. CMS/Proxy$_O$ adds the Dcs-Redirect header giving the information about this call redirection.

(6) INVITE:	***Description***
INVITE sip:+1-212-555-3333,lrn=212-265@Host(dp-f) ;user=np-queried SIP/2.0	
Via: SIP/2.0/UDP Host(dp-o.provider); branch = 2	
Via: SIP/2.0/UDP Host(mta-o.provider);	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1	
Dcs-Laes: Host(df-t)/Host(df-t);surveillancekey	*Surveillance information for MTA$_T$*
Dcs-Redirect: <tel:+1-212-555-2222> <tel:+1-212-555-2222> 1	*Original called party, and the party doing the most recent redirect*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
CSeq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_F$ queries the directory server to determine the IP address (MTA$_F$) associated with 212-555-3333. It then forwards the INVITE message to MTA$_F$. Included in the Dcs-State header is the additional surveillance information.

(7) INVITE:	***Description***
INVITE sip:555-3333@Host(mta-f.provider); user=phone SIP/2.0	
Via: SIP/2.0/UDP Host(dp-f.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
Dcs-Media-Authorization: 22S21718	
Dcs-State: Host(dp-f.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-f.provider):4321/22S21718; laes=full; redirect=<tel:+1-212-555-2222> <tel:+1-212-555-2222> 1; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Electronic Surveillance information, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

The subsequent signaling call flows are identical to those shown in Figure 29.

Call Transfer (Blind) with Transferer under Surveillance

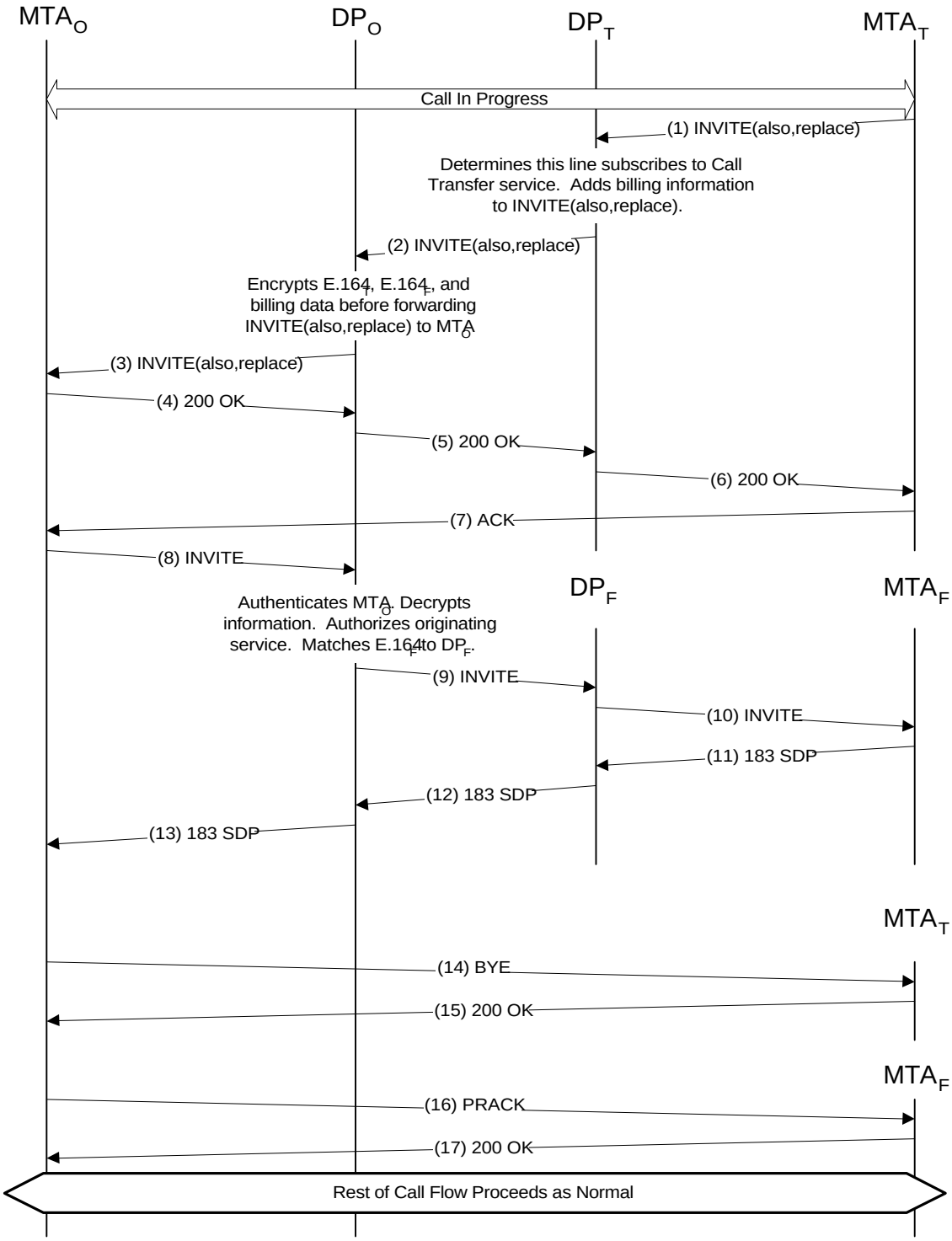


Figure 52: Call Redirection while under Surveillance

For this example, consider an existing call initiated by MTA_O to MTA_T (with MTA_T under surveillance), with the following call identification. The only difference from a normal call from MTA_O to MTA_T, as

given elsewhere in this specification, is the addition of "laess=full" in the encrypted call state information associated with CMS/Proxy$_T$.

MTA$_T$ state for call from MTA$_O$ to MTA$_T$	***Description***
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	Call leg identification
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-o.provider)	Contact address for end-to-end signaling messages
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; laes=full; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	Encrypted state information from CMS/Proxy stored in MTA
Dcs-Billing-Info: Host(rks-o.provider)/04FA37<5123-0123-4567-8900/212-555-1111/212-555-2222>	Billing Information, stored in Gate
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	Unique Billing identifier for this call

The basic call flow for call transfer is identical to that given in Appendix N, and only the differences are noted here.

MTA$_T$ desires to transfer the existing call to 555-3333 and issues an INVITE(also,replace) message, identical to (1) in Appendix N.

CMS/Proxy$_T$ decrypts the Dcs-State information and sees the "laes=full." It inserts one additional header component into the Dcs-Also header, giving the local Electronic Surveillance Delivery Function's (DF's) address information and security key to use for messages to the DF. Since surveillance was marked as "full," it adds the DF's address for both signaling information and for call content.

(2) INVITE(also,replace):	***Description***
INVITE sip: Host(dp-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(dp-t.provider)	
Via: SIP/2.0/UDP Host(mta-t.provider)	
Supported: org.ietf.sip.100rel	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	
Dcs-Also: tel:+1-212-555-3333? Dcs-Billing-Info= Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222> & Dcs-Billing-Info= Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333> & Dcs-Billing-ID= Host(dp-o.provider): 36123E5C:0152 & Dcs-Laes= Host(df-t)/Host(df-t);surveillancekey & Dcs-Redirect=<tel:+1-212-555-2222><tel:+1-212-555-2222>1	*Added Dcs-Laes header, giving the local Electronic Surveillance Delivery Functions address and security key to use for messages to the DF.* *Added Dcs-Redirect header, giving the previous redirection information*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

CMS/Proxy$_O$ includes the Dcs-Laes information in the encrypted URL given to MTA$_T$.

(3) INVITE(also,replace):	***Description***
INVITE sip: Host(mta-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(dp-o.provider), {via="Host(dp-t.provider); branch=1"; via=Host(mta-t.provider)}$_K$	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*

Dcs-Also: sip:{type=transfer; dest=tel:+1-212-555-3333; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info=Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info=Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; laes="Host(df-t)/Host(df-t);surveillancekey"; redirect=<tel:+1-212-555-2222><tel:+1-212-555-2222>1}$_K$@Host(dp-o.provider);private	*Dcs-Also: contains the laes information.*
From: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
To: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 8001 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

MTA_O acknowledges receipt of the INVITE(also,replace) by sending a 200-OK to MTA_T. This is identical to Appendix N.

After processing the INVITE(also,replace), MTA_O issues a INVITE to MTA_F. In addition to the standard headers carried in an INVITE message, the encrypted Dcs-Laes fields received in the INVITE(also,replace) message are copied into the INVITE message.

(8) INVITE:	***Description***
INVITE sip:{type=transfer; dest=tel:+1-212-555-3333; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info=Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>; billing-info=Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>; laes="Host(df-t)/Host(df-t);surveillancekey"; redirect=<tel:+1-212-555-2222><tel:+1-212-555-2222>1}$_K$@Host(dp-o.provider);private SIP/2.0	*Destination for the INVITE is taken from the Dcs-Also: header in the INVITE-REPLACE above. Private-param indicates the information is encrypted, and the first encrypted item, transfer, indicates the format.*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
Dcs-Anonymity: Off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 129 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

When the $CMS/Proxy_O$ receives the INVITE it decrypts the header information to find the real destination, and discovers the Dcs-Laes information. This is included in the INVITE message sent to $CMS/Proxy_F$. $CMS/Proxy_O$ also includes a Dcs-Redirect header giving the original destination for the call from MTA_O, the forwarding endpoint, and the number of redirections that have occurred to this call.

(9) INVITE:	***Description***
INVITE sip: +1-212-555-3333,lrn=212-265@Host(dp-f);user=np-queried SIP/2.0	
Via: SIP/2.0/UDP Host(dp-o.provider); branch=1;	
Via: SIP/2.0/UDP Host(mta-o.provider);	
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	
Dcs-Billing-Info: Host(rks-t.provider)<4278-9865-8765-9000/212-555-2222/212-555-3333>	
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1	
Dcs-Laes: Host(df-t)/Host(df-t);surveillancekey	*Surveillance information for new call*
Dcs-Redirect: <tel:+1-212-555-2222> <tel:+1-212-555-2222> 1	*Surveillance information for new call*
From: “Alien Blaster” <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	
Cseq: 129 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_F$ includes the surveillance information in its Gate-Set command, and passes the call-identifying information to its local Electronic Surveillance Delivery Function (DF_F) who passes the information on to DF_T. The encrypted Dcs-State value stored at MTA_F includes the surveillance parameters needed for possible mid-call transfers.

(10) INVITE:	***Description***
INVITE sip:555-3333@Host(mta-f.provider); user=phone SIP/2.0	
Via: SIP/2.0/UDP Host(dp-f.provider), {via=”Host(dp-o.provider); branch=1”; via=Host(mta-o.provider)}$_K$	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	
Dcs-Media-Authorization: 31S14621	
Dcs-State: Host(dp-t.provider); state=”{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; laes=full; redirect=<tel:+1-212-555-2222><tel:+1-212-555-2222>1; state=”Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0”}$_K$”	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Electronic Surveillance information, and all previous state headers from other proxies*
From: “Alien Blaster” <sip:B64(SHA-1(555-1111; time=36123E98; seq=74))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E98; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E98;seq=74))@localhost	

Cseq: 129 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Remainder of the call is identical to the basic call flow shown in Appendix N and in Figure 29, and is not repeated here.

Call Transfer with New Destination Unable to Perform Interception

If CMS_F determines that it is unable to perform the required surveillance, is passes the request back to CMS_O. This occurs, for instance, if the new destination of this redirected call is a voicemail server, or a announcement server, or a bridge server, or any other network server that does not implement the capability to intercept the media packets. CMS_F includes the Dcs-Laes header in the 183-Session-Progress message as follows:

(12) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-f.provider); nexthop=sip:555-3333Host(mta-f.provider); gate=Host(cmts-f.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=1	
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
Dcs-Anonymity: off	
Dcs-Laes: Host(df-o)/Host(df-o);surveillancekey	*Surveillance information for new call*
From: “Alien Blaster” <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

$CMS/Proxy_O$ performs the surveillance in its Gate-Set command to its CMTS. Remainder of the call flow is identical to a normal call.

Call Transfer (Consultative) with Transferer under Surveillance

After some period of consultation, MTA_O initiates a transfer of the call from MTA_{T1} to the new destination, MTA_{T2}. This involves placing the second call on hold (message sequence described earlier), and sending an INVITE(also,replace) message to MTA_{T2}, giving it the information about the call with MTA_{T1} in the Also: header. The INVITE message, since it changes parties involved in the call, is routed through the proxies. The sequence is shown in Figure 43, and detailed below.

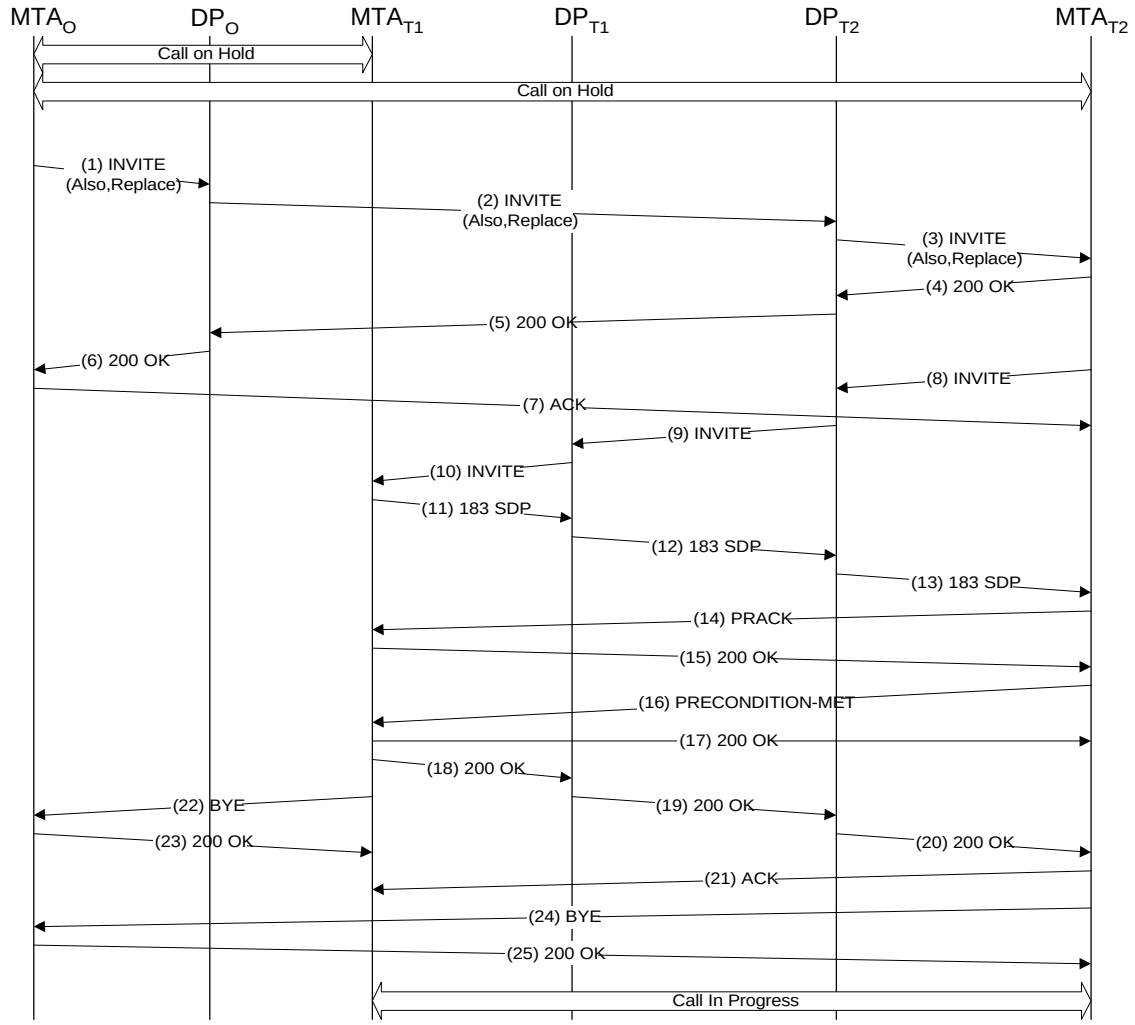


Figure 53: Call Transfer w/Consultation while under Surveillance

For this example, consider a call initiated by MTA_{T1} to MTA_X, with MTA_X under surveillance, where MTA_X performed a blind transfer to MTA_O. The call between MTA_{T1} and MTA_O is therefore under surveillance. MTA_O desires to transfer the call (with consultation) to MTA_{T2}. After placing the call to MTA_{T2}, and placing that call on hold, MTA_O initiates a transfer by sending an INVITE(also,replace) to MTA_{T2}, routed through the proxies.

The basic call flow for consultative transfer is identical to that given in Appendix O, and only the differences due to surveillance are noted here.

(1) INVITE(also,replace):	***Description***
INVITE sip: Host(mta-t2.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
Dcs-Anonymity: off	
Dcs-Also: tel:+1-212-555-2222 ? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111 & Dcs-State= Host(dp-o.provider); state="{nexthop=sip:Host(dp-t1.provider); gate=Host(cmts-o.provider):3612/17S30124; laes=full; redirect=<tel:+1-212-555-7777><tel:+1-212-555-1111>2; state="Host(dp-t1.provider); nexthop=sip:555-2222@Host(mta-t1.provider); gate=Host(cmts-t1.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*Identifies new call leg to be created. The URL is from the Dcs-Remote-Party-ID header of call from mta-t1. Call-ID gives the Call-ID to be used for the new call mta-t2 makes to mta-t1. Use of the same Call-ID as MTA$_O$'s call with mta-t1 causes the transfer. Dcs-State included due to matching Call-ID and Dcs-Replaces matching To: header value of call from mta-t1 to mta-o. Note this Dcs-State header includes laes and redirect information.*
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/3S10782, nexthop=sip:+1-212-555-3333,lrn=212-256@Host(dp-t2.provider), state="Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State information of CMS/Proxy, encrypted and stored in MTA*
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	

CMS/Proxy$_O$ decrypts the Dcs-State information in the Dcs-Also header to determine the local state information. CMS/Proxy$_O$ inserts billing information and Electronis Surveillance information into the Dcs-Also header. CMS/Proxy$_O$ then forwards the message to CMS/Proxy$_{T1}$.

(2) INVITE(also,replace):	***Description***
INVITE sip: Host(dp-o.provider) SIP/2.0	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	
Dcs-State: Host(dp-t2.provider); nexthop=sip:555-3333@Host(mta-t2.provider); gate=Host(cmts-t2.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	
Dcs-Also: tel:+1-212-555-2222? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111 & Dcs-Billing-Info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> & Dcs-Billing-Info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333> & Dcs-Billing-ID= Host(dp-o.provider): 36123E5C:0152 & Dcs-Laes=Host(df-o)/Host(df-o);securitykey & Dcs-Redirect=<tel:+1-212-555-7777><tel:+1-212-555-1111>2	*Original billing information will be used for a pseudo-call from originator to the point where the call was forwarded, and new billing information used for a pseudo-call from forwarding location to the new destination. Original billing identifier is kept for the forwarded call* *Surveillance information for the original call form MTA$_{T1}$ is retained*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	

CMS/Proxy$_{T2}$ forwards the INVITE(also,replace) message to MTA$_{T2}$ after encrypting the destination of the transfer, and the Electronic Surveillance headers.

(3) INVITE(also,replace):	***Description***
INVITE sip: 555-3333@Host(mta-t2.provider) SIP/2.0	*Routing information obtained from Dcs-State header value*

Via: SIP/2.0/UDP Host(dp-t2.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Also: sip:{type=transfer; dest=tel:+1-212-555-2222; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> ; billing-info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>; laes=Host(df-o)/Host(df-o);securitykey & redirect=<tel:+1-212-555-7777><tel:+1-212-555-1111>2}$_K$@Host(dp-t2.provider);private ? Call-ID=B64(SHA-1(555-1111;time=36124033;seq=72) & Dcs-Replaces=tel:555-1111	*Dcs-Also: contains the encrypted forwarder, new destination, Billing-identifier, timestamp, and Billing-Information fields. All are checksummed, signed by CMS/Proxy$_{T2}$, and encrypted.*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	
From: sip:B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	*Call leg identification*
To: tel:555-3333	
Call-ID: B64(SHA-1(555-1111;time=36124125;seq=23))@localhost	
CSeq: 133 INVITE	
Dcs-Replaces: sip:B64(SHA-1(555-1111;time=36124125; seq=23))@localhost	

MTA_{T2} acknowledges receipt and understanding of the INVITE(also,replace) by sending a 200-OK to MTA_O. This message is routed through the CMS/Proxy CMS/Proxy$_{T2}$, CMS/Proxy$_O$, and then delivered to MTA_O. MTA_O responds directly with an ACK. CMS/Proxy$_O$ is now done, while MTA_O is waiting for the BYE message, which will come after MTA_{T2} contacts the new destination.

After processing the INVITE(also,replace), MTA_{T2} issues a INVITE to MTA_{T1}. In addition to the standard headers carried in an INVITE message, the encrypted {Dcs-Laes, Dcs-Redirect} fields received in the INVITE(also,replace) message are copied into the Request-URI of the INVITE message. These fields indicate the surveillance information.

(8) INVITE:	***Description***
INVITE sip:{type=transfer; dest=tel:+1-212-555-2222; billing-id=Host(dp-o.provider): 36123E5C:0152; expires=<timestamp>; billing-info= Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111> ; billing-info= Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>; laes=Host(df-o)/Host(df-o);securitykey & redirect=<tel:+1-212-555-7777><tel:+1-212-555-1111>2}$_K$@Host(dp-t2.provider);private SIP/2.0	*Destination for the INVITE is taken from the Dcs-Also: header in the INVITE-REPLACE above. Private-param indicates the information is encrypted, and the first encrypted item, transfer, indicates the format.*
Via: SIP/2.0/UDP Host(mta-t2.provider)	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Smith <tel:555-3333>	
Dcs-Anonymity: Off	
From: "Alien Blaster" <sip:B64(SHA-1(555-3333; time=36124172; seq=74))@localhost>	
To: sip:B64(SHA-1(555-3333; time=36124172; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124033;seq=72)@localhost	
Cseq: 129 INVITE	
Dcs-Replaces: tel:555-1111	
Contact: sip:Host(mta-t2.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t2.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	

a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

When the CMS/Proxy$_{T2}$ receives the INVITE it first decrypts the header information to find the real destination for the call, and the surveillance information.

(9) INVITE:	***Description***
INVITE sip: +1-212-555-2222,lrn=212-265@Host(dp-t1);user=np-queried SIP/2.0	
Via: SIP/2.0/UDP Host(dp-t2.provider); branch=1;	
Via: SIP/2.0/UDP Host(mta-t2.provider);	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-3333>	
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-t2.provider):3612/17S30124/37FA1948	
Dcs-Billing-Info: Host(rks-t1.provider)<4278-9865-8765-9000/212-555-2222/212-555-1111>	
Dcs-Billing-Info: Host(rks-t2.provider)<5123-0123-4567-8900/212-555-1111/212-555-3333>	
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
Dcs-Laes: Host(df-o)/Host(df-o);securitykey	*Laes header give the Delivery Function address for sending intercepted signaling and data*
Dcs-Redirect: <tel:+1-212-555-7777><tel:+1-212-555-1111>2	*Redirect header gives the transfer history of this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-3333; time=36124172; seq=74))@localhost>	
To: sip:B64(SHA-1(555-3333; time=36124172; seq=75))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36124033;seq=72)@localhost	
Cseq: 129 INVITE	
Dcs-Replaces: tel:555-1111	
Contact: sip:Host(mta-t2.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, CMS/Proxy$_{T1}$ queries the directory server to determine the IP address (MTA$_{T1}$) associated with 212-555-2222. It then forwards the INVITE message to MTA$_{T1}$, after stripping off all of the billing fields, and adding the encrypted state information.

Remainder of the call completes as shown in Appendix O.

Appendix Y Operator Services Call Flow

Operator services calls are just like normal call flows, with the destination being an OSPS system. The CMS/Proxy performs its usual translation of the 0+ destination, and routes these calls to a PSTN gateway, where they complete normally. Operator assisted dialing, collect calling, and person-to-person, are all handled by OSPS as separate calls originated by OSPS and bridged by the OSPS. Requests for time and charges likewise is bridged by the OSPS so the operator can rejoin the call when it completes. There is no impact on the Distributed Call Signaling for any of these cases.

In the future it is possible that the Operator Services will migrate to an IP-based system, and 0+ calls will be routed there instead of through a PSTN gateway. Operator assisted dialing, credit card use, and billing calls to third parties, is a simple call transfer, identical to that shown for Call Forward No Answer above. Busy line verification and Operator break-in would be performed as described above. Implementation of requests for time&charges requires further study.

Appendix Z Privacy with Application-Level Anonymizer

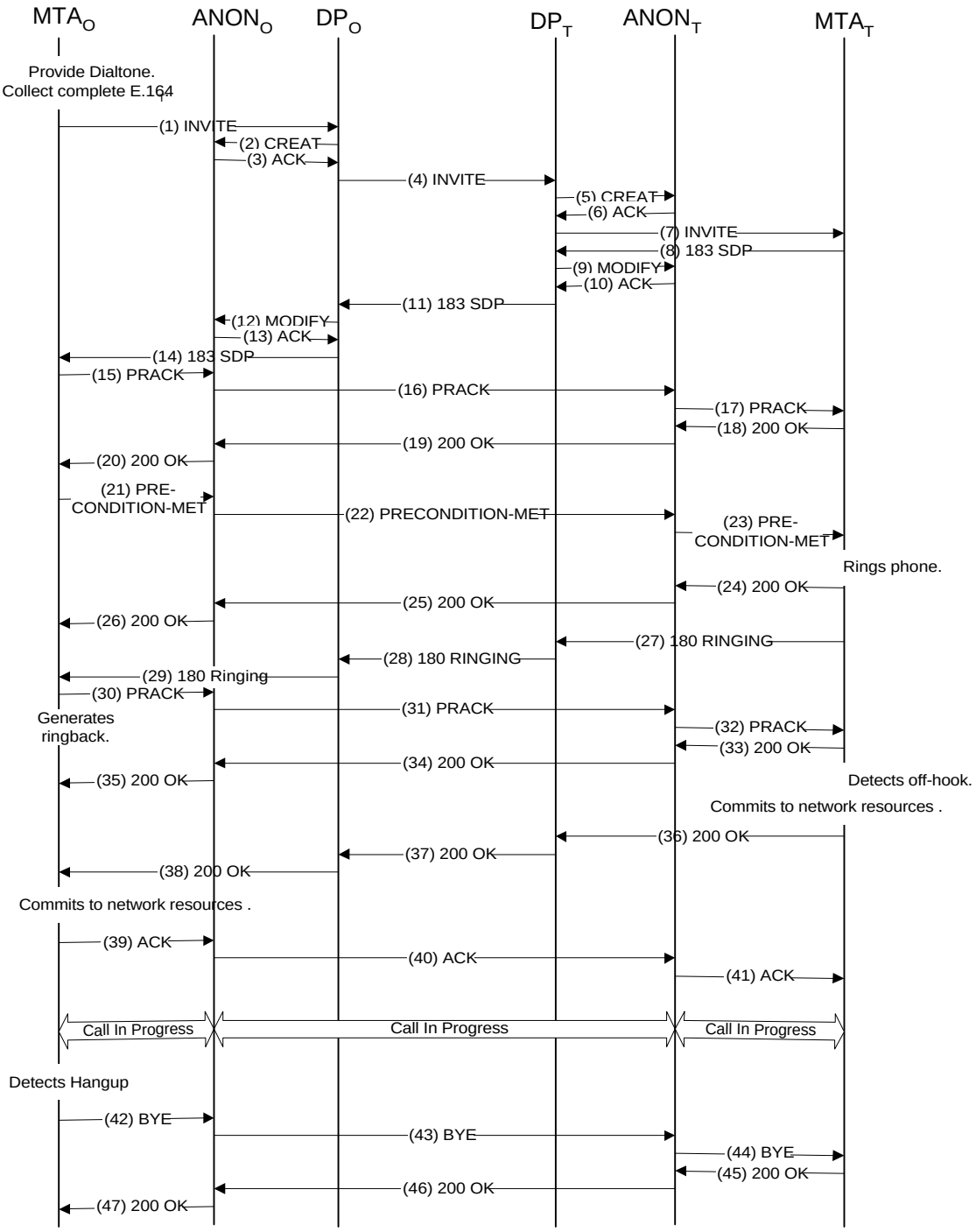


Figure 54: Application Level Anonymizer

This example only shows the case where both endpoints want privacy, which leads to an anonymizer at both ends. I think we need to think through the two other cases: (1) originator wants privacy and dest doesn't care, and (2) dest wants privacy and originator doesn't care. In both cases only one anonymizer is needed, at the far end from the endpoint wanting privacy. (So if mta-o want privacy, the only addresses mta-t will see are ann-t; if mta-t wants privacy, the only addresses mta-o will see are ann-o). Recognize also that this has inter-domain implications for the amount of trust one service provider has for another.

Case 1. Originator wants privacy and destination doesn't. Dp-t goes through same 5-6-9-10 sequence as shown in the figure. All just works OK.

Case 2. Destination want privacy but originator didn't. The un-anonymized SDP is sent to mta-t in the INVITE, but it better not use it. CMS/Proxy-o does an operation not shown here that establishes both addresses at ann-o in handling the 183. PRACK goes through ann-o, so mta-t gets the anonymized SDP here. Sounds like an additional condition under which mta-o must send the SDP in the PRACK. Mta-t still has to be smart enough to not use the initial SDP if nothing is attached to the PRACK.

The call begins identically to that shown in Appendix B of a basic MTA-originated call. The only difference at the originating MTA is that the Dcs-Anonymity header is set to "Full" or includes "IPAddr."

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(DP-o);user=phone SIP/2.0	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	
Dcs-Anonymity: Full	*Calling name and number privacy is required for this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuite:312F	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc- codecs:96	

In addition to its normal functions, CMS/Proxy$_o$ also checks the Dcs-Anonymity indication which is set to "full", so both caller-id/calling name blocking and IP address privacy must be provided. CMS/Proxy$_o$ therefore contacts the anonymizer to create an anonymous session:

(2) ANON_Create:	***Description***
ANON_Create	*Example command for setting up an anonymous session through the anonymizer.*
Endpoint1: Host(mta-o.provider):Port(mta-o.provider)	*First endpoint is mta-o:3456.*
Endpoint2:	*Second endpoint not yet known*

The anonymizer responds back with an anonymizer endpoint address:

(3) ANON_Ack:	*Description*
ANON_Ack	*Anonymizer response.*
AnonAddr: Host(ann-o.provider):Port(ann-o.provider)	*Anonymizer address for relay* Host(ann-o.provider):Port(ann-o.provider)

The anonymizer implements a packet relay and call signaling gateway between the two endpoints. The first endpoint specified in the ANON_Create will receive the anonymizer service. Any packet received for the anonymizer address specified will be forwarded to Endpoint1. Any packet sent by Endpoint1 to the anonymizer address will be forwarded to Endpoint2, but now with a source address of AnonAddr.

Having received the anonymizer address for the call, CMS/Proxy$_O$ generates the following INVITE message and sends it to CMS/Proxy$_T$. CMS/Proxy$_O$ modifies a number of parameters to the INVITE message. These are noted below.

(4) INVITE:	*Description*
INVITE sip:+1-212-555-2222,lrn=212-234@Host(dp-t);user=np-queried SIP/2.0	
Via: SIP/2.0/UDP Host(DP-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: John Doe; <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Full	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(ann-o.provider)	*Modified by CMS/Proxy$_O$ due to Anonymizer*
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-o.provider)	*Modified by CMS/Proxy$_O$ due to Anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio Port(ann-o) RTP/AVP 0	*Modified due to Anonymizer*
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

CMS/Proxy$_t$ also checks the Dcs-Anonymity indication which is set to "full", so both caller-id/calling name blocking and IP address privacy must be provided. We furthermore assume, that MTA$_t$ has requested privacy, i.e. the originating party must not be able to tell the IP-address of MTA$_t$. CMS/Proxy$_t$ therefore contacts an anonymizer to create an anonymous session:

(5) ANON_Create:	*Description*

ANON_Create	*Example command for setting up an anonymous session through the anonymizer.*
Endpoint1:	*First endpoint not yet known (specifically the port)*
Endpoint2: Host(ann-o.provider):Port(ann-o.provider)	*Second endpoint is the originating anonymizer*

The anonymizer responds back with an anonymizer endpoint address:

(6) ANON_Ack:	***Description***
ANON_Ack	*Anonymizer response.*
AnonAddr: Host(ann-t.provider):Port(ann-t.provider)	*Anonymizer address for relay* Host(ann-t.provider):Port(ann-t.provider)

The anonymizer implements a packet relay and call signaling gateway between the two endpoints. The first endpoint specified in the ANON_Create will receive the anonymizer service. Any packet received for the anonymizer address specified will be forwarded to Endpoint1. Any packet sent by Endpoint1 to the anonymizer address will be forwarded to Endpoint2, but now with a source address of AnonAddr.

CMS/Proxy$_T$ now generates the following INVITE message and sends it to MTA$_T$.

(7) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA$_T$.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	
Dcs-Remote-Party-ID: <sip:{type=remote-id; orig=tel:+1-212-555-1111; anonymity=full}$_K$@Host(dp-t.provider);private>; rpi-id=private	*Encrypted URL to maintain privacy of caller*
Dcs-Media-Authorization: 31S14621	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(ann-t.provider):Port(ann-t.provider)	*Modified by CMS/Proxy$_T$ due to anonymizer*
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-t.provider)	*Modified by CMS/Proxy$_T$ due to anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio Port(ann-t.provider) RTP/AVP 0	*Modified by CMS/Proxy$_T$ due to anonymizer*
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE, MTA$_T$ authenticates that the message came from CMS/Proxy$_T$ using IPSec. MTA$_T$ checks the telephone line associated with the E.164$_T$ to see if it is available. If it is available, MTA$_T$

looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA_T stores the INVITE message, including the encrypted Dcs-State parameters, for later use. MTA_T puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to $CMS/Proxy_T$. MTA_T starts timer (T-proxy-response).

(8) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	*Via headers as presented in INVITE message, telling nothing of call routing to MTA_T.*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Called party name and number, as provided by MTA*
Dcs-Anonymity: full	*Privacy requested by destination also*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	*Address for future direct signaling messages to MTA_T*
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, $CMS/Proxy_t$ updates the anonymizer with the MTA_t address information:

(9) ANON_Modify:	***Description***
ANON_Modify	*Example command for setting up an anonymous session through the anonymizer.*
Endpoint1: Host(mta-t.provider):Port(mta-t.provider)	*First endpoint is the terminating MTA*
Endpoint2: Host(ann-o.provider):Port(ann-o.provider)	*Second endpoint is the originating anonymizer*

The anonymizer responds back:

(10) ANON_Ack:	***Description***
ANON_Ack	*Anonymizer response.*

Following the anonymizer update, $CMS/Proxy_T$ then forwards the following 183-Session-Progress message to $CMS/Proxy_O$, restoring the Via headers, and adding Dcs-Gate information. At this point $CMS/Proxy_T$ has completed its transaction and does not maintain any more state for this call, processing all further signaling messages as a stateless proxy.

(11) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	*Authenticated id of called party*
Dcs-Anonymity: full	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(ann-t.provider):Port(ann-t.provider)	*Modified by CMS/Proxy$_T$ due to anonymizer*
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-t.provider)	*Modifed by CMS/Proxy$_T$ due to anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-t.provider) RTP/AVP 0	*Modified by CMS/Proxy$_T$ due to anonymizer*
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_O$ informs the originating anonymizer about the second endpoint:

(12) ANON_Modify:	***Description***
ANON_Modify	*Example command for modifying anonymous session through the anonymizer.*
Endpoint1: Host(mta-o.provider):Port(mta-o.provider)	*First endpoint is mta-o.*
Endpoint2: Host(ann-t.provider) :Port(ann-t.provider)	*Second endpoint is anonymizer ann_t*

The anonymizer responds back:

(13) ANON_Ack:	***Description***
ANON_Ack	*Anonymizer response.*

Subsequently, CMS/Proxy$_o$ forwards the following 183-Session-Progress to MTA$_O$. At this point CMS/Proxy$_O$ has completed its transaction and does not maintain any more state for this call, processing all further signaling messages as a stateless proxy.

(14) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	
Dcs-Media-Authorization: 17S30124	

Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(DP-t), state="Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0"}$_K$"	*State blob encrypted with a CMS/Proxy$_O$ private key containing: E.164$_O$; E.164$_T$; CMTS$_O$ IP address:port and Gate-ID, and routing to destination MTA*
Dcs-Remote-Party-ID: <sip:{type=remote-id; orig=tel:+1-212-555-2222; anonymity=full}$_K$@Host(dp-t.provider);private>; rpi-id=private	*Encrypted URL to maintain privacy of caller*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(ann-o.provider):Port(ann-o.provider)	*Modified by CMS/Proxy$_O$ due to anonymizer*
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-o.provider)	*Modified by CMS/Proxy$_O$ due to anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-o) RTP/AVP 0	*Modified by CMS/Proxy$_O$ due to anonymizer*
a-X=pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, MTA$_O$ sends the following PRACK message indirectly to MTA$_T$ through the anonymizer using the IP address in the Contact header of the 183-Session-Progress message.

(15) PRACK:	***Description***
PRACK sip:Host(ann-o.provider) SIP/2.0	*Address from Contact: line of 183-Session-Progress message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a-X-pc-qos:mandatory sendrecv	

ANN$_O$ modifies the address information in the message and forwards it to ANN$_T$.

(16) PRACK:	***Description***

PRACK sip:Host(ann-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	
Rack: 9021 127 INVITE	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-o.provider)	*Modified by Anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-o.provider) RTP/AVP 0	*Modified by Anonymizer*
a-X-pc-qos:mandatory sendrecv	

ANN_T modifies the address information in the message and forwards it to MTA_T.

(17) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-t.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	
Rack: 9021 127 INVITE	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-t.provider)	*Modified by Anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-t.provider) RTP/AVP 0	*Modified by Anonymizer*
a-X-pc-qos:mandatory sendrecv	

MTA_T acknowledges the PRACK with a 200-OK, and begins to reserve the resources necessary for the call.

(18) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

ANN_T modifies the address information in the message and forwards it to ANN_O.

(19) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	

ANN_O modifies the address information in the message and forwards it to MTA_O.

(20) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	

After sending PRACK(7), MTA_O attempts to reserve network resources if necessary. If resource reservation is successful, MTA_O sends the following PRECONDITION-MET message directly to MTA_T. MTA_O starts timer (T-direct-request).

(21) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(ann-o.provider) SIP/2.0	*Address from Contact: line of 183-Session-Progress message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (...)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

ANN_O modifies the address information in the message and forwards it to ANN_T.

(22) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(ann-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	

Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-o.provider)	*Modified by Anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-o.provider) RTP/AVP 0	*Modified by Anonymizer*
a=X-pc-qos:success sendrecv	

ANN_T modifies the address information in the message and forwards it to MTA_T.

(23) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(mta-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-t.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(ann-t.provider)	*Modified by Anonymizer*
b=AS:64	
t=907165275 0	
a=X-pc-csuites:312F	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio Port(ann-t.provider) RTP/AVP 0	*Modified by Anonymizer*
a=X-pc-qos:success sendrecv	

MTA_T acknowledges the PRECONDITION-MET message with a 200-OK.

(24) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

ANN_T modifies the address information in the message and forwards it to ANN_O.

(25) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	

ANN_O modifies the address information in the message and forwards it to MTA_O.

(26) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(26), MTA_O stops timer (T-direct-request).

Upon receipt of the (17) PRACK message, MTA_T stops timer (T-proxy-response) and attempts to reserve network resources if necessary. Once MTA_T both receives the PRECONDITION-MET message and has successfully reserved network resources, MTA_T begins to send ringing voltage to the designated line and sends the following 180 RINGING message through $CMS/Proxy_T$. MTA_T restarts the session timer (T3) with value (T-ringing).

(27) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored in MTA_T for this session.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

$CMS/Proxy_T$ decodes the Via: headers, and passes the 180-Ringing to $CMS/Proxy_O$. This operation is done as a SIP stateless proxy.

(28) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(ann-t.provider):Port(ann-t.provider)	*Modified by $CMS/Proxy_T$*
Cseq: 127 INVITE	
RSeq: 9022	

$CMS/Proxy_O$ handles the message as a SIP stateless proxy, and passes the 180-Ringing to MTA_O.

(29) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(ann-o.provider):Port(ann-o.provider)	*Modified by CMS/Proxy$_O$ due to anonymizer*
Cseq: 127 INVITE	
RSeq: 9022	

Upon receipt of the 180 RINGING message, MTA_O restarts the transaction timer (T3) with value (T-ringing). MTA_O acknowledges the provisional response with a PRACK, and plays audible ringback tone to the customer.

(30) PRACK:	***Description***
PRACK sip:Host(ann-o.provider) SIP/2.0	*Address from Contact: line of 200-OK message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
RAck: 9022 127 INVITE	*Message being acknowledged*

ANN_O modifies the address information in the message and forwards it to ANN_T.

(31) PRACK:	***Description***
PRACK sip:Host(ann-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	
RAck: 9022 127 INVITE	

ANN_T modifies the address information in the message and forwards it to MTA_T.

(32) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-t.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	
RAck: 9022 127 INVITE	*Message being acknowledged*

MTA_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(33) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*

To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

ANN_T modifies the address information in the message and forwards it to ANN_O.

(34) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	

ANN_O modifies the address information in the message and forwards it to MTA_O.

(35) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	

Once MTA_T detects off-hook on the called line, it disconnects ringing voltage from the line and sends the final response through the proxies. MTA_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, MTA_T may also commit to resources that have been reserved for this call. At this point, MTA_T begins to generate bearer channel packets of encoded voice and send them to MTA_O using the IP address and port number specified in the SDP part of the original INVITE message.

(36) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(dp-o.provider); branch=1"; via=Host(mta-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621; state="Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0"}$_K$"	*State information stored in MTA_T for this session.*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Proxy_T$ handles the message as a SIP stateless proxy, and forwards it to $CMS/Proxy_O$.

(37) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*

To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

CMS/Proxy$_O$ handles the message as a SIP stateless proxy, and forwards it to MTA$_O$.

(38) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, MTA$_O$ stops timer (T-ringing) and stops playing audible ringback tone to the customer and begins to play the bearer channel stream that is received from MTA$_T$. MTA$_O$ sends the following ACK message to MTA$_T$. If necessary, MTA$_O$ may also commit to resources that have been reserved for this call. At this point, MTA$_O$ begins to generate bearer channel packets of encoded voice and send them to MTA$_T$ using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(39) ACK:	***Description***
ACK sip:Host(ann-o.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

ANN$_O$ modifies the address information in the message and forwards it to ANN$_T$.

(40) ACK:	***Description***
ACK sip:Host(ann-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

ANN$_T$ modifies the address information in the message and forwards it to MTA$_T$.

(41) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-t.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, MTA$_T$ stop timer (T-proxy-response).

When either MTA detects hangup, it sends out a BYE message to the other MTA. In this example, MTA$_O$ detected that the customer hung up the phone. MTA$_O$ puts that line in the "idle" state so new calls can be

made or received. It sends the following BYE message directly to MTA_T. MTA_O may also need to release network resources that have been used for the call. MTA_O starts timer (T-direct-request).

(42) BYE:	***Description***
BYE sip:Host(ann-o.provider) SIP/2.0	*Address from Contact: header of previous message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

ANN_O modifies the address information in the message and forwards it to ANN_T.

(43) BYE:	***Description***
BYE sip:Host(ann-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

ANN_T modifies the address information in the message and forwards it to MTA_T.

(44) BYE:	***Description***
BYE sip:Host(mta-t.provider) SIP/2.0	*Modified by Anonymizer*
Via: SIP/2.0/UDP Host(ann-t.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of the BYE message, MTA_T stops playing the bearer channel stream received from MTA_O and, if necessary, releases network resources that have been used for this call. MTA_T sends the following 200-OK message to MTA_O. MTA_T starts a 15-second timer (T-hangup) (Note: this is a local interface issue, and not part of this specification). If MTA_T does not detect hangup on the line before timer (T-hangup) expires, it plays "reorder" tone on the customer line. Once hangup is detected, MTA_T puts that line in the "idle" state so new calls can be made or received.

(45) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-t.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

ANN_T modifies the address information in the message and forwards it to ANN_O.

(46) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(ann-o.provider)	*Modified by Anonymizer*

From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

ANN_O modifies the address information in the message and forwards it to MTA_O.

(47) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	*Modified by Anonymizer*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 131 BYE	

Upon receipt of 200-OK, MTA_O stops timer (T-direct-request).

Appendix AA Integration With Other PacketCable Specifications

In addition to their critical role in establishing end-to-end connections, the CMS/Proxies must also perform various functions to support the Quality of Service (QoS) required for those connections. This section gives a few examples of the additional procedures done by the Gate Controllers, which are explained in more detail in [4].

Establishing the proper quality of service for a voice connection through a data network requires two steps, here referred to as *reserve* and *commit*. After an initial message exchange between the two parties, which establishes both their mutual desire to communicate, and the detailed resource requirements needed for that communication, both parties must make a request for those resources. In the segmented resource allocation model of PacketCable, each party must request the resources needed on their access network, and both request the resources for a one-way path through the backbone network. Resources are merely *reserved* at this point, not given, and may be used by the network for other short-term traffic but may not be reserved by others. Once the destination party answers the ringing phone, resources are *committed* to this conversation, at which point usage billing starts. The call signaling protocols must provide opportunities for the endpoints to do this resource allocation at the proper times and with the proper information.

Prevention of theft of service requires that resource requests be authorized by a trusted network entity prior to their being allocated. It is therefore necessary for the CMS/Proxies to provide authorization for quality of service to the appropriate network resource management. The call signaling protocols must provide opportunities for the CMS/Proxies to do this authorization at the proper times and with the proper information.

Basic Call Flow– (MTA to MTA) Integration with Dynamic Quality of Service

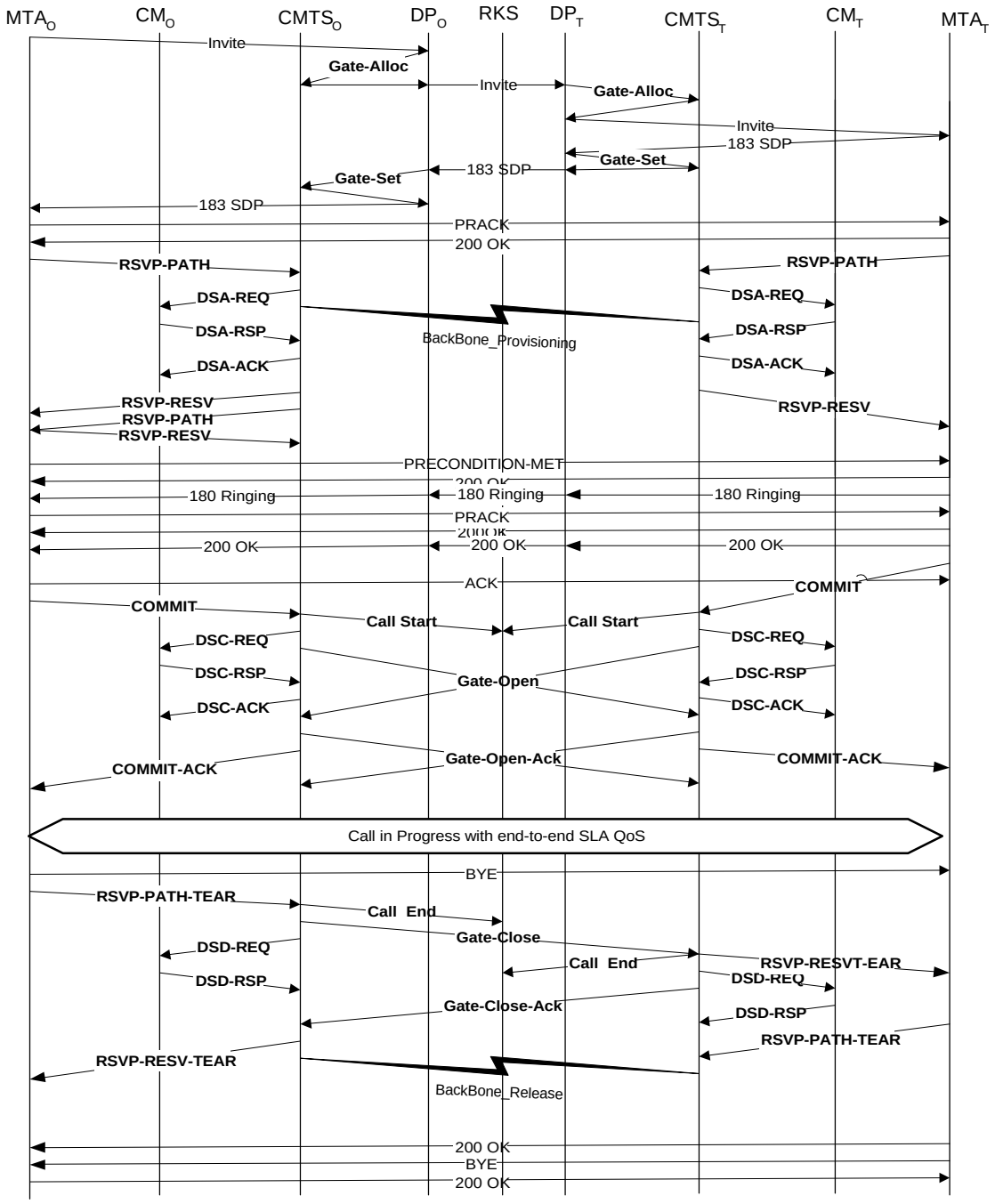


Figure 55: Basic Call Flow with QoS Messages

This is an informational, informal description of the relationship between the DCS call signaling protocol and the Dynamic QoS methods that may be invoked at different points in the basic call flow. This description is not meant to be complete. While we attempt to be accurate here in this section, the DCS call signaling specification overrides this description for the specification of the call signaling flows. When an

INVITE message is issued from the originating MTA and arrives at the CMS/$PROXY_O$, then CMS/$PROXY_O$ issues a GATE-ALLOC request to the CMTS closest ($CMTS_O$) to the originating MTA. This is a request for the allocation of a 32-bit GateID that is unique within that CMTS. This GateID is communicated to the remote CMTS in the INVITE message that is forwarded by the CMS/$PROXY_O$. In addition, the originating CMTS communicates the number of active connections (gates) that are used by MTA_O to avoid any denial of service attacks that MTA_O may launch. When the INVITE arrives at the terminating proxy, CMS/$PROXY_T$, a GATE-ALLOC request is made to the CMTS closest ($CMTS_T$) to the terminating MTA. The terminating $CMTS_T$ allocates a local GateID, and returns this to CMS/$PROXY_T$.

When the terminating proxy receives the 183 Session Progress message, the terminating CMS/$PROXY_T$ sets up the Gate at the terminating CMTS (using a GATE-SET exchange), and also provides the GateID allocated at the originating CMTS that was available from the INVITE, and is returned in the 183 Session Progress message. At this time, the CMS/$PROXY_T$ knows all the possible codecs that may be used in the call, and can check the authorization provided in the INVITE by the originating CMS/$PROXY_O$ as well as the local information it has about the terminating MTA_T. The GATE-SET message includes the "Authorized Envelope" of the Flowspec parameters. This will be subsequently used by the terminating $CMTS_T$ to admit a reservation request.. When the 183 Session Progress is returned from terminating MTA_T, included in that is the GateID allocated by the terminating CMTS. This is provided to the originating CMTS in the corresponding GATE-SET exchange. Also provided is the "Authorized Envelope" of Flowspec parameters from the CMS/PROXY to the CMTS. Upon receiving the 183 at CMS/$PROXY_O$, a GATE-SET exchange with $CMTS_O$ results in a corresponding Gate with the "Authorized Envelope" being set up at $CMTS_O$.

When the 183 Session Progress returns to the originating MTA_O, the address of the destination MTA_T is known. Also, the parameters associated with the call (codecs used) are known, and can be translated to Flowspec parameters for both directions. The originating MTA_O sends out a PRACK for the 183 and awaits a 200 OK final response for the PRACK. On receipt of the 200 OK, MTA_O is now free to perform a reservation. When the PRACK arrives at the terminating MTA_T, the MTA is allowed to perform a reservation. The terminating MTA_T knows the IP address and all the information necessary to make a reservation for the call. It is free to issue a reservation anytime now.

Reservation involves issuing a RSVP_PATH message with Flowspec parameters for both directions. This reservation process occurs at both the originating and terminating ends, as shown in the figure. The CMTS performs admission control, after checking the parameters against both the Authorized Envelope as well as resource availability, and acknowledges successful reservation with a RSVP_RESV message. Prior to the transmission of the RSVP_RESV message, the DOCSIS 1.1 Dynamic Service Addition for the layer 2 resources is performed by the corresponding CMTS. We call the Envelope of parameters in the RSVP_PATH and RSVP_RESV messages as the "Admitted Envelope". The resources required for the call are now ready to be activated. However, they await one more phase of the call signaling protocol, and for the users on both ends of the call picking up the "phone" to communicate before activating the resources.

Once MTA_O has successfully made its reservation, it sends a PRECONDITION-MET message to MTA_T implying a command to alert the far-end user (ring the destination telephone). If MTA_T successfully reserved the resources needed for the call, it responds with both a 200 OK message acknowledging the PRECONDITION-MET message that was received as well as a 180 Ringing message to indicate that the terminating phone is ringing, and that the calling party should be given a ringback call progress tone. The originating MTA_O responds with another Provisional ACK (PRACK) to acknowledge receipt of the 180 Ringing message and generates local ringback call progress tone on the originating end. The terminating MTA_T responds to the PRACK with a 200 OK to acknowlege the PRACK. When the called party answers, by going off-hook, MTA_T sends a 200-OK final response that flows through the proxies, which MTA_O acknowledges. This 200-OK final response from MTA_T to the originating MTA_O via the proxies is an indication that the two users (in this simple 2-party basic call) are ready to communicate At this point the resources that were previously reserved may be committed to this conversation. . The terminating MTA is able to send a "COMMIT" message immediately after sending the 200-OK. The originating MTA on receiving the 200-OK acknowledges this message and issues a COMMIT message also. The COMMIT message going from either MTA to its local CMTS causes a DOCSIS 1.1 Dynamic Service Change (DSC)

to activate the flow. When the COMMIT is Acknowledged by the CMTS, the two ends may begin to communicate while receiving enhanced QoS. When the COMMIT message is received by the CMTS, it starts a timer that awaits reception of the GateCommit message from the remote CMTS with this GateID.

Also indicated is the Gate Coordination messages (Gate-Open and Gate-Open-Ack) between the two CMTSs indicating to each other that the Gate has been opened, and the description (FlowSpec) of the flow expected from the other end. Reception of the GateCommit message indicates that the timer at the CMTS would be disabled.

On completion of the call, the MTAs send a RSVP_PATHTEAR message to tear down the call. At this time, the CMTS also sends a GateClose coordination messages (Gate-Close and Gate-Close_Ack) to the remote CMTS.

Call Hold Call Flow Integration with QoS

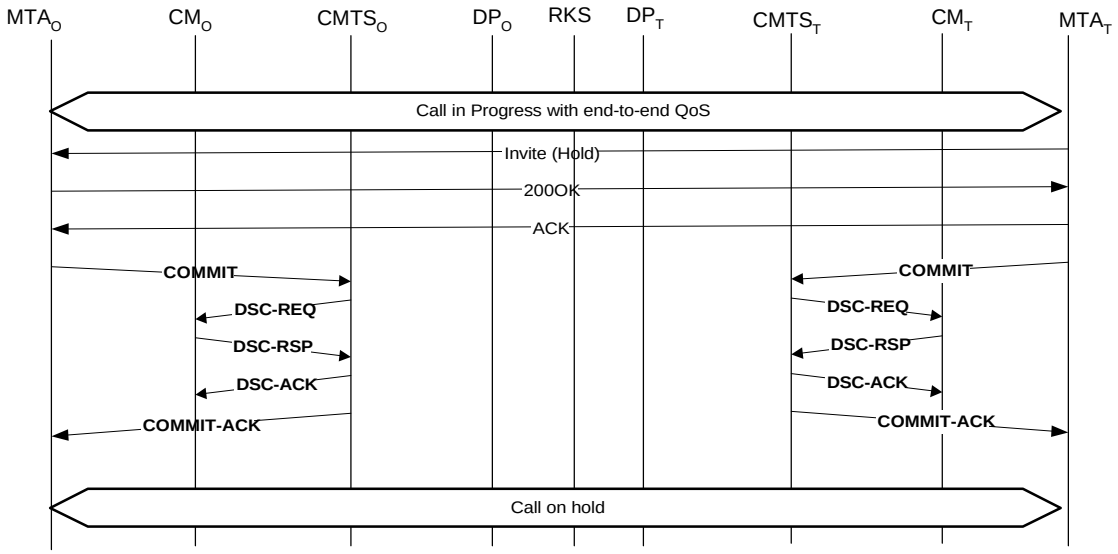


Figure 56: Call Hold Call Flow with QoS Messages

Figure 56 shows the sequence of QoS messages exchanged in order to place an existing conversation on hold. Signaling messages exchanged end-to-end establish the desire and willingness for the orginator (MTA_O) of the call to be placed on hold by the other end; each party then initiates a COMMIT exchange with their local CMTS to perform the function. The CMTS implements the COMMIT as a DOCSIS Dynamic-Service-Change operation to inform the Cable Modem of the change to the service flow.

During the period the call is on hold, the MTA must perform refresh operations (not shown) to maintain the reservation on the resources needed to re-establish the connection. When the refresh is not received at the CMTS on a timely basis, the call will be torn down.

Call Waiting Call Flow Integration with QoS

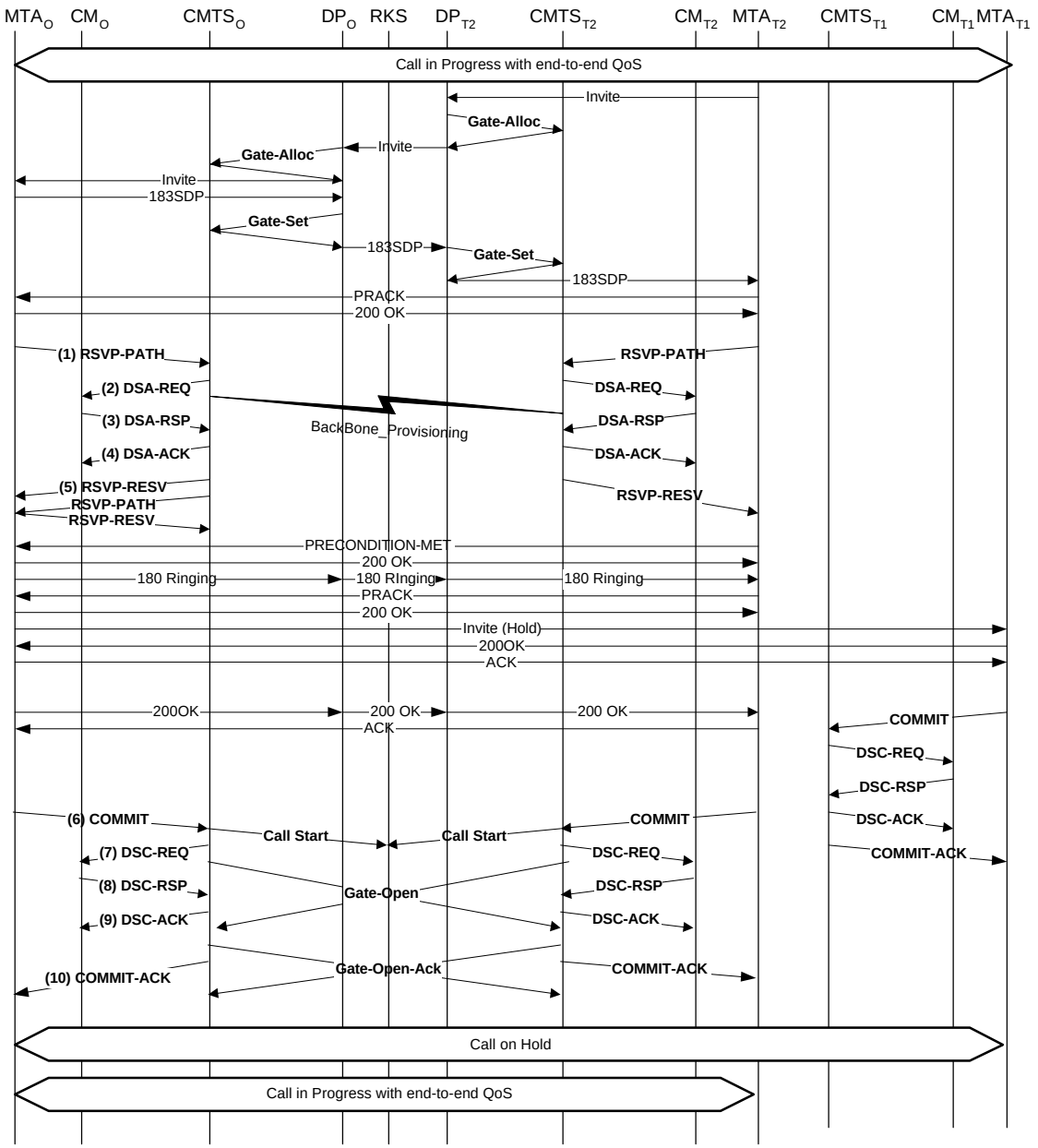


Figure 57: Call Waiting Call Flow with QoS Messages

Call Waiting is a service that allows a customer to respond to an incoming call during the time the end-point is already on an existing call. With existing standard telephone sets, the customer hears an audible alerting tone, and indicates acceptance of the new call via a hookflash (putting the previous call on hold). Subsequent hookflashes switch between the two active calls. The originator of the second call MAY hear a distinctive ringback tone.

For this example, consider an existing call initiated by MTA_O to MTA_{T1} Consider a new call intiated by MTA_{T2} . The initial set of messages associated with the second arriving call, as shown in Figure 57, are very similar to those involved in a Basic Call Setup. In response to the INVITE for the second incoming

call, the user at MTA_O is provided some indication of the second call, e.g. using a special tone. The QoS for the path between MTA_O and MTA_{T2} is reserved as shown in the Figure and the PRECONDITION-MET message is sent from MTA_{T2} to MTA_O, eliciting a 200 OK response back acknowledging it from MTA_O. Resources are reserved in the path from MTA_O and MTA_{T2}. Within the DQoS framework, by using the same Resource-ID, the access resources for the new call leg from MTA_O can be the same as the access resources for the call between MTA_O and MTA_{T1}. Subsequently, a 180 Ringing/PRACK/200 OK exchange between the two MTAs takes place. At this point, we are ready to switch the call to the new user, if the user at MTA_O hits a flash hook in response to the alerting tone. MTA_O issues a INVITE(Hold) message to MTA_{T1} to put it on HOLD. The 200 OK final response from MTA_{T1} contains an updated SDP description for the stream to be received at MTA_O, indicating an IP address of 0.0.0.0 for a held call. MTA_{T1} responds to the 200-OK message with the standard SIP ACK message. At this point it is safe for MTA_O to stop sending voice payload packets to MTA_{T1} and not risk dropping the connection due to "dead MTA recovery." The access resources for the call leg from MTA_{T1} may be "de-comitted" (but not made available for other calls of the same priority.) Thus, MTA_{T1} goes through a COMMIT-COMMIT-ACK exchange with its local CMTS, taking advantage of the Dynamic Service Change primitives in DOCSIS 1.1 to de-activate the access resources allocated for this call.

Once the first conversation is successfully placed on hold, MTA_O indicates a completion to the "ringing" to MTA_{T2}. MTA_O issues a 200-OK to MTA_{T2}. This 200 OK is routed through proxies. The corresponding ACK completes the call signalling. Both MTA_O and MTA_{T2} go through the process of committing their access resources and the call between them is in progress. The access resources for MTA_O would be the same as that used for the original call leg. At this point the user at MTA_{T2} has a connection to the second caller, , with the first caller, MTA_1, on hold.

Subsequent hookflashes repeat the sequence of INVITE(hold)/200-OK/ACK to one destination, and INVITE(resume)/200-OK/ACK to the other. Once the 200-OK is received, it is safe for MTA_O to stop sending voice packets. INVITE (Resume) is very similar, except that the SDP description includes the proper IP address in the "c=" line. MTA_{T1} acknowledges the Resume command with a 200-OK message. The response contains an updated SDP description for the stream to be received at MTA_{T1}, indicating the real IP address of Host(mta-T1.provider). MTA_O responds to the 200-OK message with the standard SIP ACK message. At this point it is safe for MTA_O to start sending voice payload packets to MTA_{T1}. Because the access resources for the call at MTA_{T1} were not released, there is no need to reserve resources, nor is there a need to perform admission control when resuming the call. The only thing MTA_{T1} needs to do is to re-commit the access resources.By the use of the same Resource-ID, MTA_O can re-use the same access resources as it switches from one call to the other.

Basic Call (MTA to PSTN) - Integration with TGCP

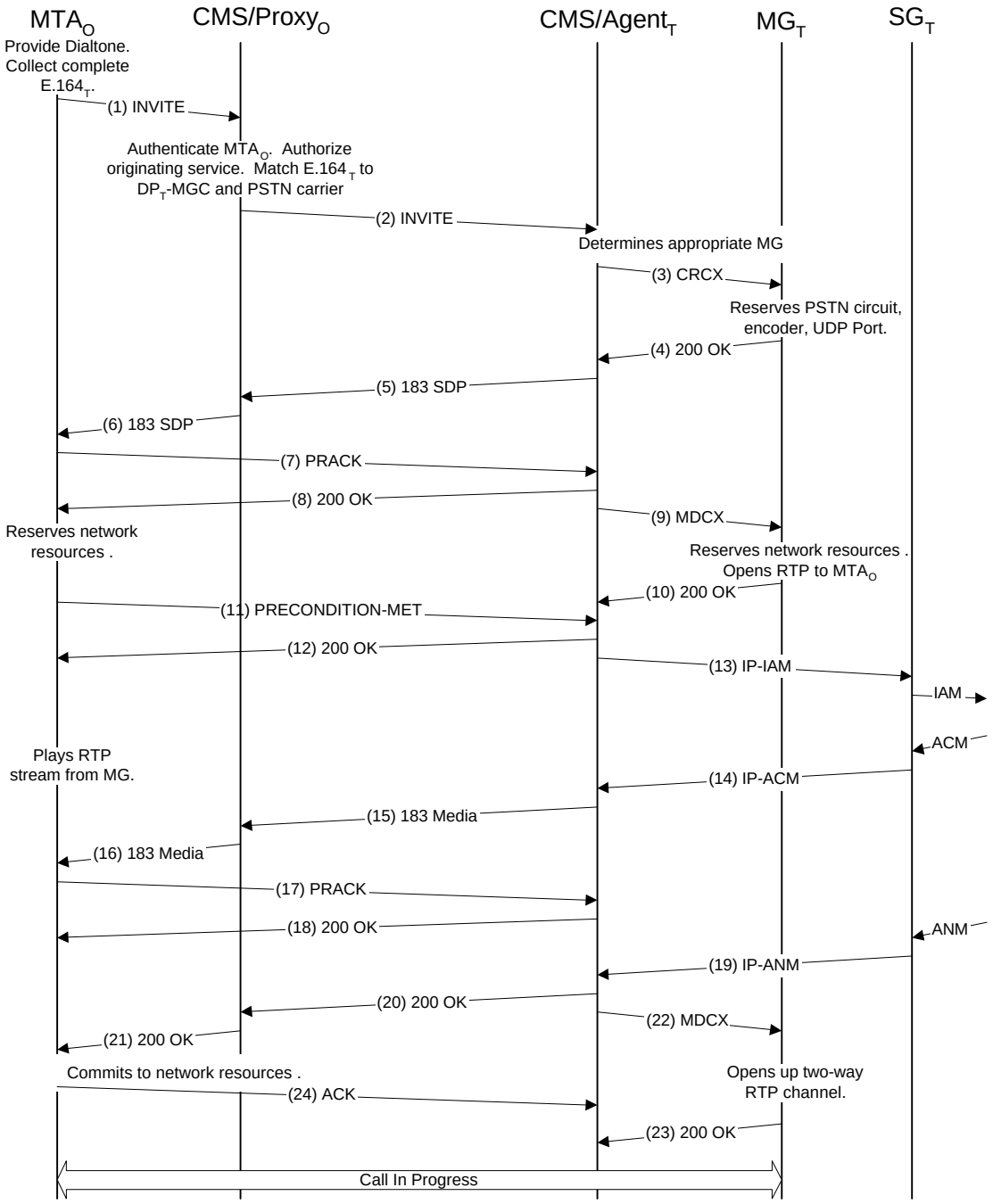


Figure 58: MTA to PSTN Signaling Call Flow

The following sections describe DCS call signaling flows for basic calls that terminate or originate on the PSTN. This is accomplished via an interworking function implemented in the Media Gateway Controller that understands DCS signaling on one side, and the Media Gateway Control Protocol (MGCP) on the other.

A call originating at an MTA begins identically to an on-net call, as described in Figure 29. For this example, we assume the same sequence as given in Figure 29, only that the destination E.164 number is located on the PSTN. The call setup begins when MTA_O detects off-hook on one of its lines. MTA_O first puts that line in the "busy" state. MTA_O sends an audible dialtone signal to the customer and begins to detect DTMF digits. Upon receiving the first digit, MTA_O stops dialtone. Once a complete E.164 number has been received (based upon a digit map that has been provisioned in the MTA), MTA_O generates the following SIP INVITE message and sends it to $CMS/Proxy_O$ (the CMS/Proxy that manages MTA_O). MTA_O starts the retransmission timer (T-proxy-request).

(1) INVITE:	***Description***
INVITE sip:555-2222@Host(DP-o);user=phone SIP/2.0	*Request URI starts with the dialed number from the user*
Via: SIP/2.0/UDP Host(mta-o.provider)	*IP Address or Domain name of originating MTA.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:555-1111>	*Calling name and number, as provided by MTA*
Dcs-Anonymity: Off	*Calling name and number privacy is not required for this call*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) uniquely identifies the call-leg, excluding the display-name in the From: header.. To maintain privacy, the addr-spec is encrypted and calling-number and calling-name will be omitted from MTA-MTA signaling.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	*To: is a cryptographical hash of a string that contains the dialed digits from the user, timestamp, and a sequence number, or other random string.*
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	*Call-ID is a cryptographically random identifier.*
Cseq: 127 INVITE	*Call sequence number*
Contact: sip:Host(mta-o.provider)	*Signaling address of originator*
Content-Type: application/sdp	*A SIP INVITE message must contain a SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP description contains lines giving the following: Version number (v= line), Connection information at originator (c= line), and Media encoding parameters and port number (m= line)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving the INVITE message, $CMS/Proxy_O$ authenticates MTA_O using standard IPSec authentication. $CMS/Proxy_O$ examines the Dcs-Remote-Party-ID: line and checks to see that this originating phone number belongs to MTA_O, and is authorized for originating service. $CMS/Proxy_O$ also checks to make sure the calling name in the Dcs-Remote-Party-ID: line is a valid calling name for this line. $CMS/Proxy_O$ then sends the dialed number to a directory server for resolution to an IP address. In this example, the directory server returns the address of CMS/MGC_T, the CMS/MGC that manages a Media Gateway (MG) which can complete the call to the PSTN. $CMS/Proxy_O$ generates the following INVITE message and sends it to CMS/MGC_T. $CMS/Proxy_O$ adds a number of parameters to the INVITE message, which are described below. Upon sending this INVITE message, $CMS/Proxy_O$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(2) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(dp-t);user=np-queried SIP/2.0	*"lrn" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(DP-o.provider);branch=1	*CMS/Proxy$_O$ IP address; branch indicates this is the first destination attempt*
Via: SIP/2.0/UDP Host(mta-o.provider)	
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: John Doe <tel:+1-212-555-1111>	*Verified Calling Name, and full E.164 Calling Number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(cmts-o.provider):3612/17S30124/37FA1948 required	*IP addr of CMTS, ID of the originating gate, and key for gate coord. Also the indication that gate coordinationis required for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing*
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by CMS/Proxy$_O$ for handling of messages from MTA$_T$ to MTA$_O$*
Dcs-Billing-ID: Host(dp-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/Proxy$_O$ IP address:timestamp:sequence#*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg. The display-name is not part of the call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mta-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*Suggested encryption key inserted by CMS/Proxy-o*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	
a=X-pc-codecs:96	

Upon receiving this INVITE message, CMS/MGC$_T$ authenticates that the sender was CMS/Proxy$_O$ using IPSec, and sends the E.164$_T$ address to the directory server. In this example, the Directory Server determines the E.164$_T$ address can be reached via MG$_T$ (one of the MGs managed by CMS/MGC$_T$) so CMS/MGC$_T$ sends a TGCP CreateConnection command to MG$_T$:

(3) CreateConnection:	***Description***
CRCX 2001 ds/ds1-1/6@Host(mg-t) MGCP 1.0 TGCP 1.0	*Trunk desired is ds/ds1-1/6@Host(mg-t)*
C: A3C47F21456789F0	*Call-Idenifier*
L: p:10, a:PCMU, sc-rtp: 62/51, sc-rtcp: 62/51	*LocalConnectionOptions specifying use of PCMU encoding and security services. Key is provided in the SDP.*
M: inactive	*Connection mode is inactive*
v=0	*CMS/MGC$_T$ removes the "a=X-pc-qos" attribute (optional, MG$_T$ would simply ignore it)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	

a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
a=rtpmap:96 G726-32/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-codecs:96	

MG_T reserves the PSTN circuit, local resources and UDP port and sends a TGCP 200 OK response back to $CMS/MGC_{T:}$

(4) 200 OK:	***Description***
200 2001 OK	*200 OK indicates success*
I: 32F345E2	*The connection-ID assigned by MG_T*
v=0	*SDP contains the MG_T bearer channel IP address, and negotiated voice encoding parameters.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

Upon receiving the TGCP 200 OK message, CMS/MGC_T sends the following 183-Session-Progresss message to $CMS/Proxy_O$, restoring the Via headers, and adding Dcs-Gate information. CMS/MGC_T also starts the retransmission timer with value (T-proxy-response) and the session timer (T3) with value (T-resource). Note that CMS/MGC_T elected not to include state information but rather maintain it itself. CMS/MGC_T may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(5) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124; orig-dest=tel:+1-212-555-2222; num-redirects=0	*State information wanted by $CMS/Proxy_O$ for handling of messages from CMS/MGC_T to MTA_O*
Dcs-Gate: Host(mgc-t.provider):4321/31S14621/37FA1948	*IP address of the terminating gate (CMS/MGC_T IP address), Gate-ID, and security key to enable gate-coordination in Dynamic QoS*
Dcs-Remote-Party-ID: <tel:+1-212-555-2222>	*Address of the called party.*
Dcs-Anonymity: off	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mgc-t.provider)	*CMS/MGC_T also serves as the destination for endpoint signaling.*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*CMS/MGC_T adds the "a=X-pc-qos" attribute.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	

c= IN IP4 Host(mg-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_O$ forwards the following message to MTA$_O$. This message contains a Dcs-State parameter giving all the information needed by the CMS/Proxy for later features. The Dcs-State value is signed by CMS/Proxy$_O$ and encrypted by CMS/Proxy$_O$'s privately-held key. At this point CMS/Proxy$_O$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy.

(6) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	
Dcs-Media-Authorization: 17S30124	*ID of gate at originator end of connection*
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(mgc-t) }$_K$"	*State blob encrypted with a CMS/Proxy$_O$ private key containing: E.164$_O$; E.164$_T$; CMTS$_O$ IP address:port and Gate-ID, and routing to destination*
Dcs-Remote-Party-ID: <tel:+1-212-555-2222>	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mgc-t.provider)	
Content-Type: application/sdp	
Content-length: (…)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a-X=pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, MTA$_O$ stops timer (T-proxy-request) and sends the following PRACK message directly to CMS/MGC$_T$ using the IP address in the Contact header of the 183-Session-Progress message.

(7) PRACK:	***Description***
PRACK sip:Host(mgc-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	

Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (…)	
v=0	*SDP description of final negotiated media stream.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	

CMS/MGC_T stops the retransmission timer (T-proxy-response) and acknowledges the PRACK with a 200-OK:

(8) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

CMS/MGC_T now instructs MG_T to reserve network resources and to modify the connection to enable media flowing from the PSTN back to MTA_o in support of PSTN generated ringback:

(9) ModifyConnection:	***Description***
MDCX 2002 ds/ds1-1/6@Host(mg-t) MGCP 1.0 TGCP 1.0	
C: A3C47F21456789F0	
I: 32F345E2	
M: sendonly	*Enable media from the PSTN to be sent to MTA_o*
<resource reservation>	*Details of TGCP network resource reservation currently unspecified.*
v=0	*SDP description of final negotiated media stream.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	*CMS/MGC_T removed the "a=X-pc-qos" attribute (optional, MG_T would simply ignore it)*
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	

MG_T modifies the connection to "sendonly" mode, performs the network resource reservation (details currently unspecified), and responds with a TGCP 200 OK message:

(10) 200 OK:	***Description***
200 2002 OK	*Command succeeded*
	Media parameters were unchanged, so SDP not returned this time.

After sending PRACK (7), MTA_O attempts to reserve network resources if necessary. If resource reservation is successful, MTA_O starts timer (T-direct-request) and sends the following PRECONDITION-MET message directly to CMS/MGC_T:

(11) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(mgc-t.provider) SIP/2.0	*Address from Contact: line of 183-Session-Progress message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.*
O=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 3456 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

Upon receipt of the PRECONDITION-MET message, CMS/MGC_T stops the session timer (T3) and acknowledges the PRECONDITION-MET message with a 200-OK:

(12) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(10), MTA_O stops timer (T-direct-request).

Once CMS/MGC_T has both received the PRECONDITION-MET message and learned that MG_T has successfully reserved network resources, CMS/MGC_T initiates the call setup to the PSTN by sending an IP-IAM (13) message to SG_T. In generating the IP-IAM message, CMS/MGC_T must honor the DCS-Anonymity setting from the called party by setting the Presentation Restriction Indicator. The result of Local Number Portability queries must be provided as well.

When the PSTN starts alerting the called user, an IP-ACM (14) message is received from SG_T, and CMS/MGC_T now generates a 183-Session-Progress to be sent to MTA_T to indicate that inband alerting is occurring. CMS/MGC_T now restarts the session timer (T3) with value (T-ringing) and the retransmission timer with value (T-proxy-response):

(15) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: Sip/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); state="{gate= Host(cmts-o.provider): 3612/17S30124, nexthop=sip:+1-212-555-2222;lrn=212-234@Host(mgc-t) }$_K$"	*State blob encrypted with a $CMS/Proxy_O$ private key containing: $E.164_O$; $E.164_T$; $CMTS_O$ IP address:port and Gate-ID, and routing to destination*
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9022	
Session: media	*Indicates media is now available.*
Contact: sip:Host(mgc-t.provider)	

$CMS/Proxy_O$ handles the message as a SIP stateless proxy, and passes the 183-Session-Progress to MTA_O (after suitable processing):

(16) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: Sip/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9022	
Session: media	*Indicates media is now available.*
Contact: sip:Host(mgc-t.provider)	

Upon receipt of the 183-Session-Progress message, MTA_O restarts the transaction timer (T3) with value (T-ringing). MTA_O acknowledges the provisional response with a PRACK, and starts receiving the inband ringback and playing it to the user.

(17) PRACK:	***Description***
PRACK sip:Host(mgc-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
Rseq: 9022 127 INVITE	*Message being acknowledged*

CMS/MGC_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response).

(18) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once CMS/MGC_T receives the IP-ANM (19) from SG_T indicating the call has been answered, CMS/MGC_T sends a TGCP ModifyConnection to MG_T to make the connection full duplex:

(22) ModifyConnection:	***Description***
MDCX 2002 ds/ds1-1/6@Host(mg-t) MGCP 1.0 TGCP 1.0	
C: A3C47F21456789F0	
I: 32F345E2	
M: sendrecv	*Enable full duplex media*

MG_T modifies the connection to "sendrecv" mode and responds with a TGCP 200 OK message:

(23) 200 OK:	***Description***
200 2002 OK	*Command succeeded*
	Media parameters were unchanged, so SDP not returned this time.

Simultaneusly with (22), CMS/MGC_T stops the session timer (T-ringing), starts timer (T-proxy-response) and sends a DCS 200 OK to indicate answer to CMS/$Proxy_o$

(20) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-o.provider);branch=1	
Via: SIP/2.0/UDP Host(mta-o.provider)	
Dcs-State: Host(dp-o.provider); nexthop=sip:555-1111@Host(mta-o.provider); gate=Host(cmts-o.provider):3612/17S30124	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

CMS/$Proxy_O$ handles the message as a SIP stateless proxy, and forwards it to MTA_O (after suitable processing):

(21) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

Upon receipt of the 200-OK message, MTA_O stops timer (T-ringing) and stops playing audible ringback tone to the customer and begins to play the bearer channel stream that is received from MTA_T. MTA_O

sends the following ACK message to CMS/MGC_T. If necessary, MTA_O may also commit to resources that have been reserved for this call. At this point, MTA_O begins to generate bearer channel packets of encoded voice and sends them to MG_T using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE).

(24) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

Upon receipt of the ACK message, CMS/MGC_T stops timer (T-proxy-response).

The call is now established.

Basic Call (PSTN to MTA) - Integration with TGCP

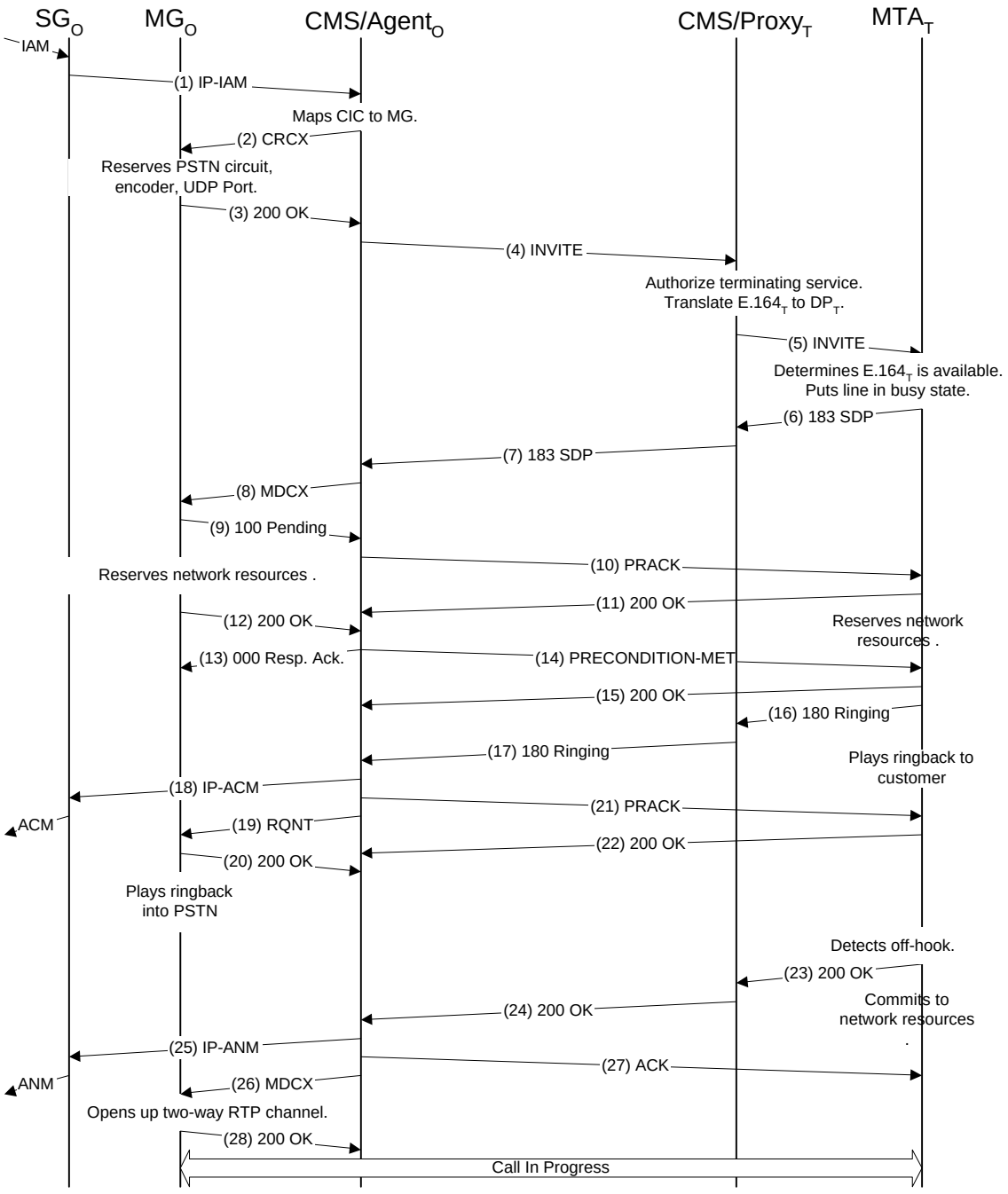


Figure 59: PSTN to MTA Signaling Call Flow

This example shows a call originating on the PSTN and directed to a destination on the PacketCable network. We assume the same sequence of user behavior as in the basic call flow of Figure 29, only difference being the location of the originator.

The call setup begins when CMS/Agent$_O$ receives an IP-IAM (1) from SG$_O$ indicating a new call to be setup. The IP-IAM indicates the trunk being used and CMS/Agent$_O$ determines that this trunk is served by MG$_O$. CMS/Agent$_O$ therefore sends a TGCP CreateConnection command to MG$_O$ to reserve the trunk as well as the MG resources needed, e.g. codec and UDP port:

(2) CreateConnection:	***Description***
CRCX 2001 ds/ds1-1/6@Host(mg-o) MGCP 1.0 TGCP 1.0	*Trunk desired is ds/ds1-1/6@Host(mg-o)*
C: A3C47F21456789F0	*Call-Idenifier*
L: p:10, a:PCMU, sc-st: clear:WhenInTheCourseOfHumanEvents, sc-rtp: 62/51, sc-rtcp: 62/51	*LocalConnectionOptions specifying use of PCMU encoding and security services..*
M: inactive	*Connection mode is inactive*

MG$_O$ carries out the request and responds with a TGCP 200 OK message including SDP for the newly created connection:

(3) 200 OK:	***Description***
200 2001 OK	*200 OK indicates success*
I: 32F345E2	*The connection-ID assigned by MG$_O$*
v=0	*SDP contains the MG$_O$ bearer channel IP address, and negotiated voice encoding parameters.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

CMS/Agent$_O$ then extracts the destination number received in the IP-IAM ("212-555-2222" in this case) and sends it to a directory server for resolution to an IP address. In this example, the directory server returns the address of CMS/Proxy$_T$. CMS/Agent$_O$ then starts the retransmission timer (T-proxy-request), forms the following INVITE message and sends it to CMS/Proxy$_T$. In so doing, CMS/Agent$_O$ extracts the Calling Party Number ("212-555-1111" in this case) and provides it as the DCS-Remote-Party-ID. CMS/Agent$_O$ also examines the Presentation Restriction Indicator ("presentation allowed" in this case) and sets the DCS-Anonymity corrrespondingly. The result of any Local Number Portability queries performed by the PSTN are provided as well. Also, the Charge Number may be examined to determine who should be billed for the call.

(4) INVITE:	***Description***
INVITE sip:+1-212-555-2222,lrn=212-234@Host(dp-t);user=np-queried SIP/2.0	*"lrn" shows that LNP dip done and gives the result. Dialed number fully expanded into E.164 number*
Via: SIP/2.0/UDP Host(mgc-o.provider)	*CMS/MGC$_O$ IP address*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: <tel:+1-212-555-1111>	*Full E.164 Calling Number*
Dcs-Anonymity: Off	
Dcs-Gate: Host(mgc-o.provider):3612/17S30124/37FA1948 optional	*IP addr of MGC$_O$, ID of the originating gate, and key for gate coord. Also the indication that gate coordination is optional for this call.*
Dcs-Billing-Info: Host(rks-o.provider)<5123-0123-4567-8900/212-555-1111/212-555-2222>	*IP address and encryption key of the record keeping server for event collection, account number, originating number, and terminating number for billing.* *Note that the account number may be that of the PSTN carrier.*
Dcs-Billing-ID: Host(mgc-o.provider):36123E5C:0152	*Unique Billing ID made up of CMS/MGC$_O$ IP address:timestamp:sequence#*

From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*The triple (From, To, CallID) is used by SIP to uniquely identify a call leg.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mgc-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*"a=X-pc-qos" attributed inserted by CMS/MGC$_O$*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	

Upon receiving this INVITE message, CMS/Proxy$_T$ authenticates that the sender was CMS/Agent$_O$ using IPSec, and sends the E.164$_T$ address to the directory server. In this example, the Directory Server is able to translate E.164$_T$ to the IP address of MTA$_T$ (one of the MTAs managed by CMS/Proxy$_T$). CMS/Proxy$_T$ then checks to see if MTA$_T$ is authorized for receiving this call. CMS/Proxy$_T$ also checks the account information to determine if the originator is paying for the call or if MTA$_T$ is expected to pay. CMS/Proxy$_T$ generates the following INVITE message and sends it to MTA$_T$. The Dcs-Remote-Party-ID line appears unchanged only if the destination MTA has subscribed to caller-id service; otherwise, or if the caller had specified privacy of the caller information, the Dcs-Remote-Party-ID line would be altered. Note that the Via lines have been encrypted, maintaining the privacy of the caller. The line Dcs-State has been added, and contains all the information needed by CMS/Proxy$_T$ for any subsequent call features that may be requested. This information is signed by CMS/Proxy$_T$ and encrypted.

Upon sending this INVITE message, CMS/Proxy$_T$ starts the retransmisison timer (T-proxy-request) and starts the T3 session timer (T-proxy-setup). The retransmission timer is cancelled on receipt of the optional 100-Trying provisional response (not present in this call flow); both are cancelled on receipt of the 183-Session-Progress provisional response.

(5) INVITE:	***Description***
INVITE sip:555-2222@Host(mta-t.provider); user=phone SIP/2.0	*Local number portability information removed. Username is a string known to MTA$_T$.*
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(mgc-o.provider)" }$_K$	*Via headers are encrypted to provide calling party privacy.*
Supported: org.ietf.sip.100rel	*Indicate support for reliable provisional responses*
Dcs-Remote-Party-ID: <tel:+1-212-555-1111>	*Present only if customer subscribes to Calling Name/Caller ID*
Dcs-Media-Authorization: 31S14621	*Gate ID at the CMTS controlling resources*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(mgc-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State blob encrypted with a CMS/Proxy$_T$ privately-held key containing: nexthop routing information, CMTS$_T$ IP address:port/Gate-ID, Via headers, and all previous state headers from other proxies*
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg Identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Contact: sip:Host(mgc-o.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description of media stream to be received by MG$_O$.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	

s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	

Upon receiving this INVITE, MTA_T authenticates that the message came from $CMS/Proxy_T$ using IPSec. MTA_T checks the telephone line associated with the $E.164_T$ (as found in the Request URI) to see if it is available. If it is available, MTA_T looks at the capability parameters in the Session Description Protocol (SDP) part of the message and determines which media channel parameters it can accommodate for this call. MTA_T stores the INVITE message, including the encrypted Dcs-State parameters, for later use. MTA_T puts this line in the "busy" state (so any other call attempts are rejected until this call clears), generates the following 183-Session-Progress response, and sends it to $CMS/Proxy_T$. MTA_T starts the retransmission timer with value (T-proxy-response) and starts the session timer (T3) with value (T-resource).

MTA_T can, at its option, still accept further incoming calls and present them all to the customer. Such enhanced user interfaces for the MTA is beyond the scope of this specification. Note that MTA_T can't use the To: header field to determine the proper line, as it may be totally unrelated to the phone number at MTA_T.

(6) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(mgc-o.provider)"}$_K$	*Via headers as presented in INVITE message.*
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State information stored in MTA_T for this session.*
Dcs-Remote-Party-ID: John Smith <tel:555-2222>	*Called name and number, as provided by MTA_T*
Dcs-Anonymity: off	*Called name and number privacy is not requested for this call*
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	*Request for acknowledgement of this provisional response*
Session: qos	
Contact: sip:Host(mta-t.provider)	*Address for future direct signaling messages to MTA_T*
Content-Type: application/sdp	*The response to INVITE in SIP must contain the SDP description of the media stream to be sent to MTA_T.*
Content-length: (…)	
v=0	*SDP contains the MTA_T bearer channel IP address, and negotiated voice encoding parameters*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Proxy$_T$ forwards the following message to CMS/Agent$_O$, restoring the Via headers, and adding Dcs-Gate information. At this point CMS/Proxy$_T$ has completed all the call processing functions needed for this call, deletes its local state information, and handles all remaining messages as a stateless proxy. CMS/Proxy$_T$ may include Dcs-Billing-Information if it wishes to override the billing information that came in the INVITE (e.g. collect or toll-free call).

(7) 183-Session-Progress:	***Description***
SIP/2.0 183 Session Progress	
Via: SIP/2.0/UDP Host(mgc-o.provider)	
Dcs-State: Host(dp-t.provider); nexthop=sip:555-2222@Host(mta-t.provider); gate=Host(cmts-t.provider):4321/31S14621; orig-dest=tel:+1-212-555-1111; num-redirects=0	*State information for CMS/Proxy$_O$ included in the INVITE message*
Dcs-Gate: Host(cmts-t.provider):4321/31S14621/37FA1948	*IP address of the terminating gate (CMTS$_T$ IP address), Gate-ID, and security key to enable gate-coordination in Dynamic QoS*
Dcs-Remote-Party-ID: John Smith <tel:+1-212-555-2222>	*Authenticated id of called party*
Dcs-Anonymity: off	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	
Rseq: 9021	
Session: qos	
Contact: sip:Host(mta-t.provider)	
Content-Type: application/sdp	
Content-length: (...)	
v=0	
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv confirm	

Upon receiving the 183-Session-Progress message, CMS/Agent$_O$ stops timer (T-proxy-request) and sends a TGCP ModifyConnection to MG$_O$ instructing it to reserve network resources:

(8) ModifyConnection:	***Description***
MDCX 2002 ds/ds1-1/6@Host(mg-o) MGCP 1.0 TGCP 1.0	
C: A3C47F21456789F0	
I: 32F345E2	
<resource reservation>	*Details of TGCP network resource reservation currently unspecified.*
v=0	*CMS/MGC$_O$ removed the "X-pc-qos" attribute (optional, MG$_O$ would simply ignore it)*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mta-t.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: B5F348G2	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	

a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

MG_O initiates the request including the resource reservation which in this case does not complete immediately. MG_O therefore quickly returns a TGCP 100 provisional response to CMS/MGC_O:

(9) 100 Pending:	***Description***
100 2002 Pending	*Provisional response*
v=0	*SDP description of media stream to be received by MG_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

On receiving this response, CMS/$Agent_O$ sends the following PRACK message directly to MTA_T using the IP address in the Contact header of the 183-Session-Progress message.

(10) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*New Cseq value for this message*
Rack: 9021 127 INVITE	*Message being acknowledged*
Content-Type: application/sdp	
Content-length: (...)	
v=0	*SDP description of final negotiated media stream.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	*CMS/MGC_O includes "a=X-pc-qos" parameter*
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:mandatory sendrecv	

MTA_T acknowledges the PRACK with a 200-OK, and begins to reserve the resources necessary for the call.

(11) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mta-o.provider)	

From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 128 PRACK	*Message being acknowledged*

Meanwhile, the resource reservation at MG_O succeeds, and MG_O therefore sends a TGCP 200 OK to $CMS/Agent_O$:

(12) 200 OK:	***Description***
200 2002 OK	*Final response*
v=0	*SDP description of media stream to be received by MG_O.*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	

$CMS/Agent_O$ acknowledges the final response by sending a TGCP 000 Response Acknowledgement to MG_O:

(13) 200 OK:	***Description***
000 2002	*Response Acknowledgement*

Since resource reservation was successful, $CMS/Agent_O$ now starts timer (T-direct-request) and sends the following PRECONDITION-MET message directly to MTA_T:

(14) PRECONDITION-MET:	***Description***
PRECONDITION-MET sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183-Session-Progress message*
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification. These three fields must match those used in the initial INVITE message.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	
Content-Type: application/sdp	*INVITE message requires an SDP description of the media flow.*
Content-length: (…)	
v=0	*SDP including the final negotiated media stream description, and the indication that qos resources have been reserved.* *Note that the "a=X-pc-qos" attribute was added by CMS/MGC_O*
o=- 2987933615 2987933615 IN IP4 A3C47F2146789F0	
s=-	
c= IN IP4 Host(mg-o.provider)	
b=AS:64	
t=907165275 0	
a=X-pc-csuites-rtp: 62/51	
a=X-pc-csuites-rtcp: 62/51	
a=X-pc-spi-rtcp: A0H662B1	
a=X-pc-secret:clear:WhenInTheCourseOfHumanEvents	
a=rtpmap:0 PCMU/8000	
m=audio 6544 RTP/AVP 0	
a=X-pc-qos:success sendrecv	

MTA_T acknowledges the PRECONDITION-MET message with a 200-OK.

(15) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 129 PRECONDITION-MET	*Message being acknowledged*

Upon receipt of the 200-OK(10), CMS/MGC_O stops timer (T-direct-request).

Upon receipt of the PRACK (10) message, MTA_T stops timer (T-proxy-response) and attempts to reserve network resources if necessary. Once MTA_T both receives the PRECONDITION-MET message and has successfully reserved network resources, MTA_T begins to send ringing voltage to the designated line and sends the following 180 RINGING message through $CMS/Proxy_T$. MTA_T restarts the session timer (T3) with value (T-ringing):

(16) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(mgc-o.provider)"}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621 }$_K$"	*State information stored in MTA_T for this session.*
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Contact: sip:Host(mta-t.provider)	
Cseq: 127 INVITE	
Rseq: 9022	

$CMS/Proxy_T$ decodes the Via: headers, and passes the 180-Ringing to $CMS/Agent_O$. This operation is done as a SIP stateless proxy:

(17) 180 RINGING:	***Description***
SIP/2.0 180 Ringing	
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

$CMS/Agent_O$ sends an IP-ACM (18) to SG_O to indicate that alerting has begun. $CMS/Agent_O$ also sends a TGCP NotificationRequest command to MG_O instructing it to generate ringback tones:

(19) NotificationRequest:	***Description***
RQNT 2003 ds/ds1-1/6@Host(mg-o) MGCP 1.0 TGCP 1.0	
S: IT/rt	*Apply ringback tone signal (from the ISUP package).*
R: IT/oc, IT/of	*Look for operation complete and operation failure*
X: AB123FE0	*RequestIdentifier for the request.*

MG_O starts applying ringback tones on the trunk towards the PSTN and acknowledges the request:

(20) 200 OK:	***Description***
200 2003 OK	

Upon receipt of the 180 RINGING message, CMS/$Agent_O$ also restarts the transaction timer (T3) with value (T-ringing) and acknowledges the provisional response by sending a PRACK directly to MTA_T:

(21) PRACK:	***Description***
PRACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: line of 183 message*
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*New Cseq value for this message*
Rseq: 9022 127 INVITE	*Message being acknowledged*

MTA_T acknowledges the PRACK with a 200-OK, and stops timer (T-proxy-response):

(22) 200 OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification.*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 130 PRACK	*Message being acknowledged*

Once MTA_T detects off-hook on the called line, it sends the final 200 OK response through the proxies. MTA_T stops timer (T-ringing) and starts timer (T-proxy-response). If necessary, MTA_T may also commit to resources that have been reserved for this call. At this point, MTA_T begins to generate bearer channel packets of encoded voice and send them to MG_O using the IP address and port number specified in the SDP part of the original INVITE message.

(23) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(dp-t.provider), {via="Host(mgc-o.provider)}$_K$	
Dcs-State: Host(dp-t.provider); state="{nexthop=sip:Host(dp-o.provider); gate=Host(cmts-t.provider):4321/31S14621}$_K$"	*State information stored in MTA_T for this session.*
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

CMS/$Proxy_T$ handles the message as a SIP stateless proxy, and forwards it to CMS/$Agent_O$ (after suitable processing):

(24) 200-OK:	***Description***
SIP/2.0 200 OK	
Via: SIP/2.0/UDP Host(mgc-o.provider)	
From: <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	*Call leg identification*
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 INVITE	

On receiving the 200 OK, CMS/Agent$_O$ generates an IP-ANM (25) and sends it to SG$_O$ to indicate that the call has been answered. CMS/Agent$_O$ also sends a TGCP ModifyConnection command to MG$_O$ instructing it to stop playing ringback tones and to place the connection in send/receive mode thereby beginning to play the bearer channel stream that is received from MTA$_T$. If necessary, MG$_O$ may also commit to resources that have been reserved for this call. At this point, MG$_O$ begins to generate bearer channel packets of encoded voice and send them to MTA$_T$ using the IP address and port number specified in the SDP part of the original 183-Session-Progress message (that was a response to the original INVITE):

(26) ModifyConnection:	***Description***
MDCX 2004 ds/ds1-1/6@Host(mg-o) MGCP 1.0 TGCP 1.0	
C: A3C47F21456789F0	
I: 32F345E2	
M: sendrecv	*Connection is to be placed in full duplex mode*
X: AB123FE1	*RequestIdentifier*
S:	*No signals specified, thus ringback stops.*

CMS/Agent$_O$ also stops timer (T-ringing) and acknowledges the 200 OK (24) with an ACK:

(27) ACK:	***Description***
ACK sip:Host(mta-t.provider) SIP/2.0	*Address from Contact: header of 183 message*
Via: SIP/2.0/UDP Host(mta-o.provider)	
From: "Alien Blaster" <sip:B64(SHA-1(555-1111; time=36123E5B; seq=72))@localhost>	
To: sip:B64(SHA-1(555-2222; time=36123E5B; seq=73))@localhost	
Call-ID: B64(SHA-1(555-1111;time=36123E5B;seq=72))@localhost	
Cseq: 127 ACK	

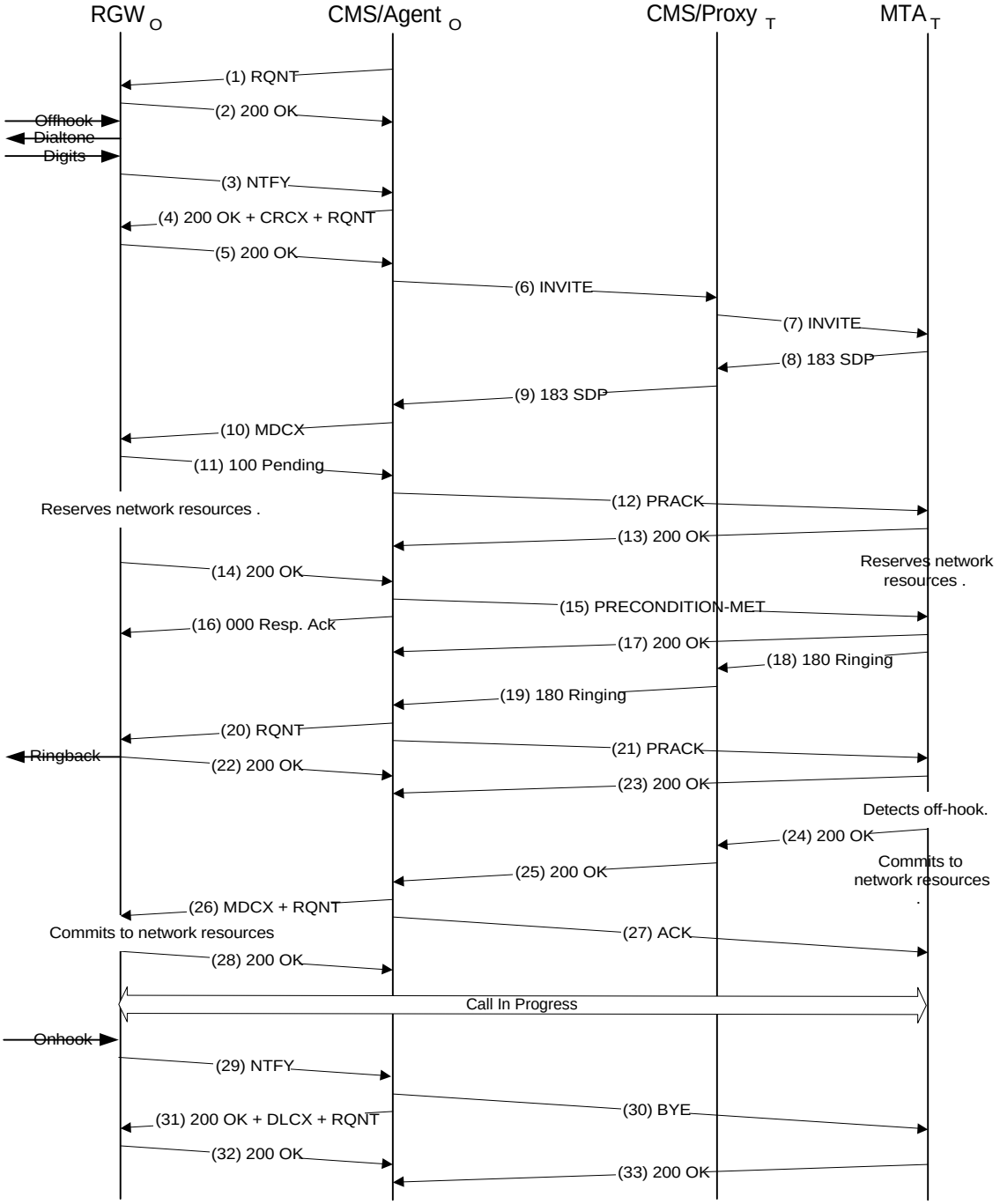
Upon receipt of the ACK message, MTA$_T$ stop timer (T-proxy-response).

When receiving the TGCP ModifyConnection (26), MG$_O$ stops applying ringback tones, places the connection in send/receive mode and acknowledges the request:

(28) 200 OK:	***Description***
200 2004 OK	

The call is now established.

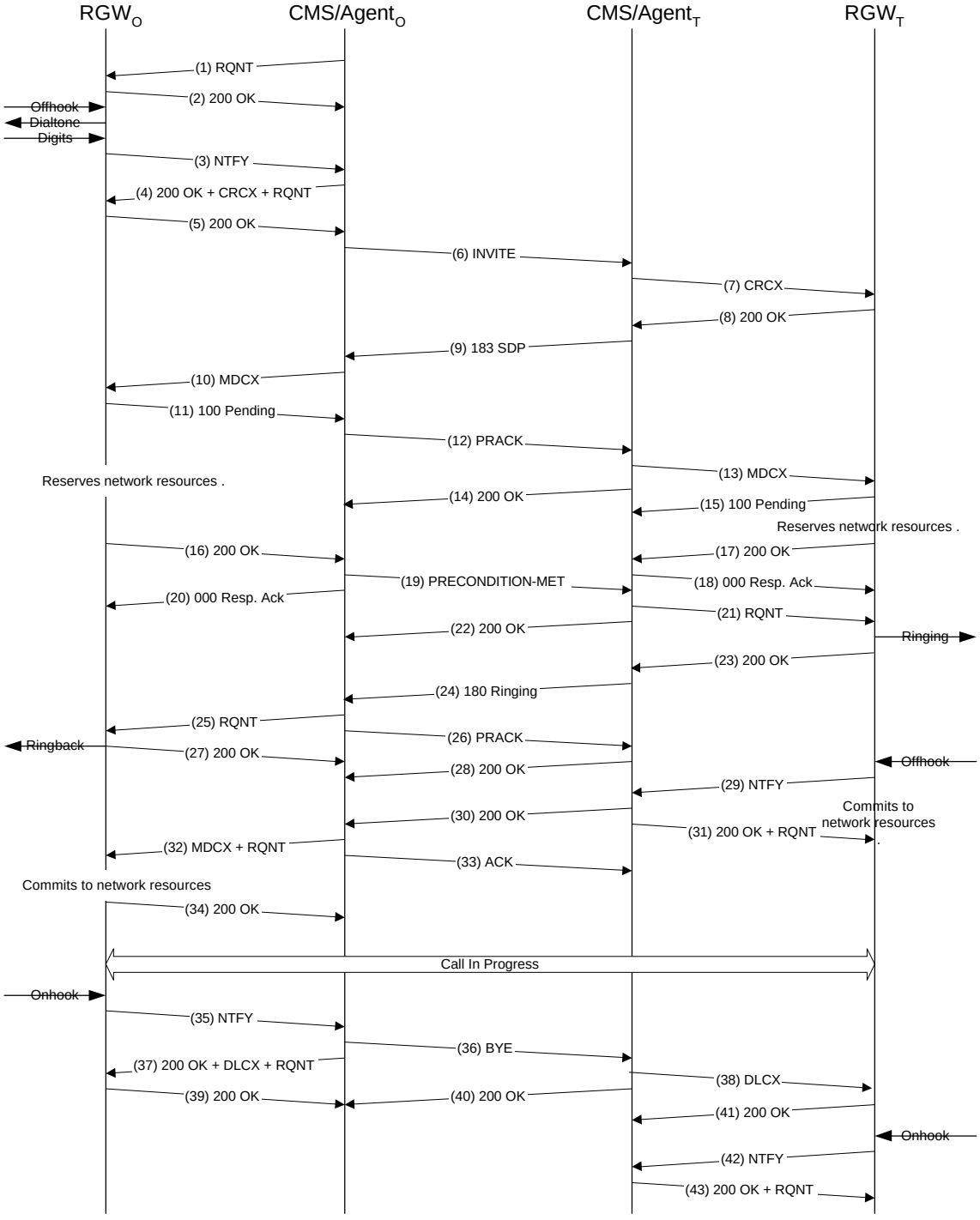
Basic Call (RGW to MTA) Interworking with Network Based Call Signaling



The Figure above shows the signaling flow for a call originated by a NCS residential gateway (RGW) and terminating at a DCS MTA.

From a DCS point of view, the flow is similar to the PSTN to MTA signaling flow shown earlier and the details will therefore be skipped here. Please refer to the NCS specification for details on NCS signaling.

Basic Call (RGW to RGW) Interworking with Network Based Call Signaling



The Figure above shows a call flow for a call between two NCS residential gateways.

From a DCS point of view, the originating part of the flow is similar to the PSTN to MTA (or RGW to MTA) signaling flow shown earlier, and the terminating part of the flow is similar to the MTA to PSTN signaling flow shown earlier, with one exception. Both CMS'es will have gate coordination as optional, and DCS-Gate information will therefore not be included in the response CMS/Agent$_T$. The details of the flow will be skipped here. Please refer to the NCS specification for details on NCS signaling.

Appendix BB Acronyms

ACR	Anonymous Call Reject
AVP	Audio Video Profile
BLV	Busy Line Verification
CA	Call Agent
CFB	Call Forwarding on Busy
CFNA	Call Forwarding No Answer
CFU	Call Forward Unconditional
CLASS	Custom Local Area Signaling Services
CMS	Call Management System
CMTS	Cable Modem Termination System
CODEC	Coder/Decoder
DCS	Distributed Call Signaling
DOCSIS	Data Over Cable Service Interface Specification
DP	DCS Proxy
EI	Emergency Interrupt
EP	Endpoint
E.164	Telephone number standard of ITU
FQDN	Fully Qualified Domain Name
GC	Gate Controller
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPSEC	IP Security
ITU	International Telecommunication Union
LAES	Lawfully Authorized Electronic Surveillance
LNP	Local Number Portability

LRN	Local Routing Number
MF	Multi-Frequency
MG	Media Gateway
MGC	Media Gateway Controller
MGCP	Media Gateway Control Protocol
MTA	Multimedia Terminal Adapter
NCS	Network Call Signaling
OSPS	Operator Services Positioning System
OSS	Operations Support System
PSTN	Public Switched Telephone Network
QoS	Quality of Service
RFC	Request for Comments (IETF standard)
RGW	Residential Gateway
RKS	Record Keeping Server
RTP	Real-Time Transport Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SS7	Signaling System #7
TCP	Transmission Control Protocol
TGCP	Trunk gateway control protocol
UAC	User agent – Client
UAS	User agent - Server
UDP	User Datagram Protocol
URI	Universal Resource Identifier
URL	Universal Resource Locator

Appendix CC Acknowledgements

This specification was developed and influenced by numerous individuals representing many different vendors and organizations. PacketCable hereby wishes to thank everybody who participated directly or indirectly in this effort. In particular, PacketCable wants to recognize the following individuals for their significant involvement and contributions to this specification: Burcak Beser, Mike Mannette, Kurt Steinbrenner (3Com); Dave Boardman (Arris), K.K. Ramakrishnan, Bill Marshall (primary author), Doug Nortz, Chuck Kalmanek, Tung-Hai Hsiao, and John Lawser (AT&T); Flemming Andreasen, Dave Oran, Bill Guckel, and Michael Ramalho (Cisco); John Pickens (Com21); Adam Roach and Anthony Toubassi (Ericsson); Javier Martinez (Lucent); Poornima Lalwaney, Jon Fellows, and John Wheeler (Motorola); Keith Kelly (NetSpeak); D.R. Evans (Secure Cable Solutions); Edward Miller, Glenn Russell, and Matt Osman (CableLabs).

Much of the text in Sections 3.2.1 and 3.3.12 came from [25], and much of the text in Sections 3.3.11 and 3.4.1 came from [16], both of which contained the following copyright notice:

Much of the text in Section 3.3.7 came from [17], which contained the following copyright notice:

Appendix DD References

PacketCable Specifications

These documents are available at http://www.packetcable.com/

[1] PacketCable Architecture Framework, pkt-tr-arch-v01-991201, December 1, 1999..

[2] PacketCable Security Specification, pkt-sp-sec-i01-991201, December 1, 1999.

[3] PacketCable Event Messaging Specification, pkt-sp-em-i01-991201, December 1, 1999..

[4] PacketCable Dynamic Quality of Service Specification, pkt-sp-dqos-i01-991201, December 1, 1999.

[5] PacketCable MTA Device Provisioning Specification, pkt-sp-prov-i01-991201, December 1, 1999.

[6] PacketCable OSS Overview, pkt-tr-oss-v01-991201, December 1, 1999.

[7] PacketCable Audio/Video Codecs Specification, pkt-sp-codec-i01-991201, December 1, 1999.

[8] PacketCable Network-Based Call Signaling Protocol Specification, pkt-sp-ec-mgcp-i02-991201, December 1, 1999.

[9] PacketCable PSTN Gateway Call Signaling Protocol Specification, pkt-sp-tgcp-i01-991201, December 1, 1999.

[10] PacketCable Electronic Surveillance Specification, pkt-sp-esp-i01-991229, December 29, 1999.

IETF RFCs

These documents are available at http://www.ietf.org/rfc/

[11] Handley, M, H. Schulzrinne, E. Schooler, J. Rosenberg, SIP: Session Initiation Protocol, RFC2543, March 1999.

[12] Handley, M, V. Jacobson, SDP: Session Description Protocol, RFC2327, April 1998.

[13] Schulzrinne, H, RTP Profile for Audio and Video Conferences with Minimal Control, RFC1890, January, 1996.

[14] Postal, J, User Datagram Protocol, RFC768, August 1980.

[15] Postal, J, Transmission Control Protocol, RFC791, September 1981.

IETF Internet-Drafts

These documents are available at http://www.ietf.org/internet-drafts/

[16] Donovan, S, SIP 183 Session Progress Message, draft-ietf-sip-183-00, October, 1999.

[17] Schulzrinne, H, J. Rosenberg, SIP Call Control Services, draft-ietf-mmusic-sip-cc-01, June, 1999.

[18] Vaha-Sipila, Antti, URLs for telephone calls, draft-antti-telephony-url-12.txt, December, 1999.

[19] Architectural Considerations for Providing Carrier Class Telephony Services Utilizing SIP-based Distributed Call Control Mechanisms, draft-dcsgroup-sip-arch-01, March, 2000.

[20] SIP Extensions for Caller Identity, Privacy, and Operator Services, draft-dcsgroup-sip-privacy-01, March, 2000.

[21] SIP Extensions for Media Authorization, draft-dcsgroup-sip-call-auth-01, March, 2000.

[22] Integration of Resource Management and SIP for IP Telephony, draft-manyfolks-sip-resource-00, March, 2000.

[23] SIP Extensions for supporting Distributed Call State, draft-dcsgroup-sip-state-01, March, 2000.

[24] SIP proxy-to-proxy extensions for supporting Distributed Call State, draft-dcsgroup-sip-proxy-proxy-01, March, 2000.

[25] Reliability of Provisional Responses in SIP, draft-ietf-sip-100rel-00, January, 2000.

[26] Rosenberg, J, Schulzrinne, H, The SIP Supported Header, draft-ietf-sip-serverfeatures-02, March 2000.

[27] Handley, M, H. Schulzrinne, E. Schooler, J. Rosenberg, SIP: Session Initiation Protocol, draft-ietf-sip-rfc2543bis-00, August 1999.