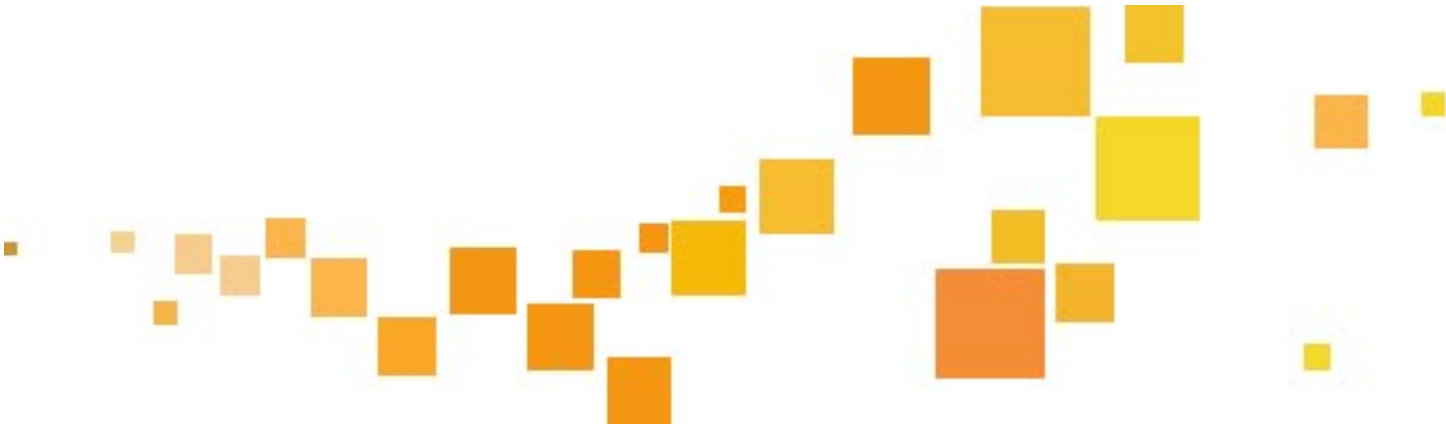
Real-Time Billing in SIP



Baruch Sterman, Ph.D.
Chief Scientist
baruch@deltathree.com

deltathree
The IP Communications Network™

Table of Contents

2 **Abstract**

3 **Review of SIP**

Introduction to SIP

SIP Network Elements

SIP Messages

SIP Responses

Why SIP

SIP Past, Present and Future

6 **SIP Call Flows**

A Simple SIP Call

Registration

Authentication

Additional SIP Capabilities

11 **Pre-Paid Billing**

Real-Time Billing Considerations

Centralized Authentication – Digest Authentication to RADIUS Mapping

Call Teardown and CDR Creation

16 **Conclusion**

17 **References**

Abstract

Session Initiation Protocol (SIP) [1] is fast becoming the protocol of choice for IP based communication – specifically telephony (Voice over Internet Protocol - VoIP), video, and instant messaging. SIP's strengths lie in its email-like addressing scheme, its plain text headers (similar to HTTP), its scalable architecture, and most of all in its ability to carry generic data (following the MIME standard of email fame). Though interconnecting a SIP call leg with the legacy phone system (Public Switched Telephone Network – PSTN) is well defined and communication flows seamlessly, the billing model regnant in the telephony world does not mesh quite so naturally with SIP. For example, the ubiquitous pre-paid calling card services require querying the account status before a call is authorized and real-time adjustment of the balance as soon as a call terminates. SIP does not address either of these functionalities. This paper will examine the requirements of such systems and suggest possible implementations within a SIP architecture.

Review of SIP

Introduction to SIP

SIP is a signaling and session management protocol that is meant to establish a communication session between two IP-based endpoints. SIP provides the capabilities to [2]:

- Find the target endpoint – Endpoints are addressed in a format similar to email, i.e. user@InternetAddress where the Internet address may be either an IP address or a fully qualified domain name, like “deltathree.com”.
- Negotiate a common method of communication – Different users may have different capabilities. In a SIP message, each user lists its own set of codecs that it supports and the precedence of each. This information is carried as SDP (Session Description Protocol) in the body of a SIP message.
- Manage the session – SIP messages are defined to set up and teardown a communication session, and can pass information between the two endpoints outside the media channel.

SIP Network Elements

The basic elements that form the building blocks of a SIP network are:

- User Agent – These are the endpoints that initiate and terminate calls. Examples of user agents are IP phones, software clients, and IP voice gateways that interconnect with the PSTN (the legacy telephone network).
- Proxy Server – Proxies forward SIP messages on to other SIP network elements according to rules that are defined within the proxy.
- Registrar – The registrar is used to dynamically track the location of user agents and forward messages to their provisional address. Typically, a user agent will “register” with a registrar and provide the information regarding its current address (i.e. IP address). The registrar will then forward calls destined for the registered user on to the specified location.
- Redirect Server – This server is similar to a proxy, but instead of forwarding messages, it responds to SIP requests with the location of a user agent or server. The entity that sent the request will issue another request directly to the location specified by the redirect server.

Often, proxy software contains proxy, registrar, and redirect functionality. A proxy may also keep a set of rules that determine how to route calls when the attempt to establish a connection with a user at its default location fails. An example of such “location service” rules would be to re-route a call to another SIP address when the primary address is busy or unavailable.

SIP Messages

The common SIP messages are:

- INVITE – Initiated by a user agent inviting a second user agent to participate in a session.
- ACK – Acknowledgement that is returned confirming that the user agent has received a final response (as opposed to a provisional response, such as “trying” or “ringing”).
- BYE – Indicates that one endpoint user agent is terminating the session.
- REGISTER – Contains the location information necessary in order to reach a user agent.
- CANCEL – Indicates that a pending request should be cancelled.

SIP Responses

SIP responses are similar to HTTP. Responses are numbered, with the first digit indicating the type of response. Table 1 shows the different response types:

Response Group	Description
1XX	Informational: Indicates status of call prior to completion
2XX	Success: Request has succeeded
3XX	Redirection: Server has returned possible locations
4XX	Client Error: The request has failed due to an error by the client
5XX	Server Failure: The request has failed due to an error by the server
6XX	Global Failure: The request has failed

Table 1: SIP Response Messages

Why SIP

SIP offers the following benefits:

- Simplicity – SIP messages are encoded in plain text and are easy to understand and troubleshoot. A SIP stack is much lighter than other VoIP stacks.
- Distributed Functionality – The proxy/forward approach utilized in SIP encourages distributing intelligence throughout the network. A call may flow from proxy to proxy, each working on the call with its own logic and functionality. Changes can be made to one component while leaving the others untouched.
- Extensibility – SIP is very flexible in that the message body can contain attachments similar to email. User agents are free to use SIP for any type of communication, as long as both endpoints can process the attached information.

SIP Past, Present and Future

SIP was initially proposed as a method of inviting users to join in large-scale multicast conferences. It was soon realized that SIP could be used to establish point-to-point IP phone calls as well. By 1999, SIP was a proposed standard for establishing communication sessions. Over the next few years, device manufacturers, software vendors, and service providers got together at “bake-offs” to test the interoperability of their SIP components. In November 2000, the 3GPP wireless initiative adopted SIP as its protocol. Further validation of SIP came in the Spring of 2001 when Microsoft announced that its Messenger platform and voice services (released in the Fall of 2001) would be SIP-based. Device manufacturers and application developers are continuing to roll out SIP based products and services.

Simple SIP Call

In order to get a better understanding of SIP and how it works, an example of a call flow between a user agent, a proxy server, and a voice gateway is described below.

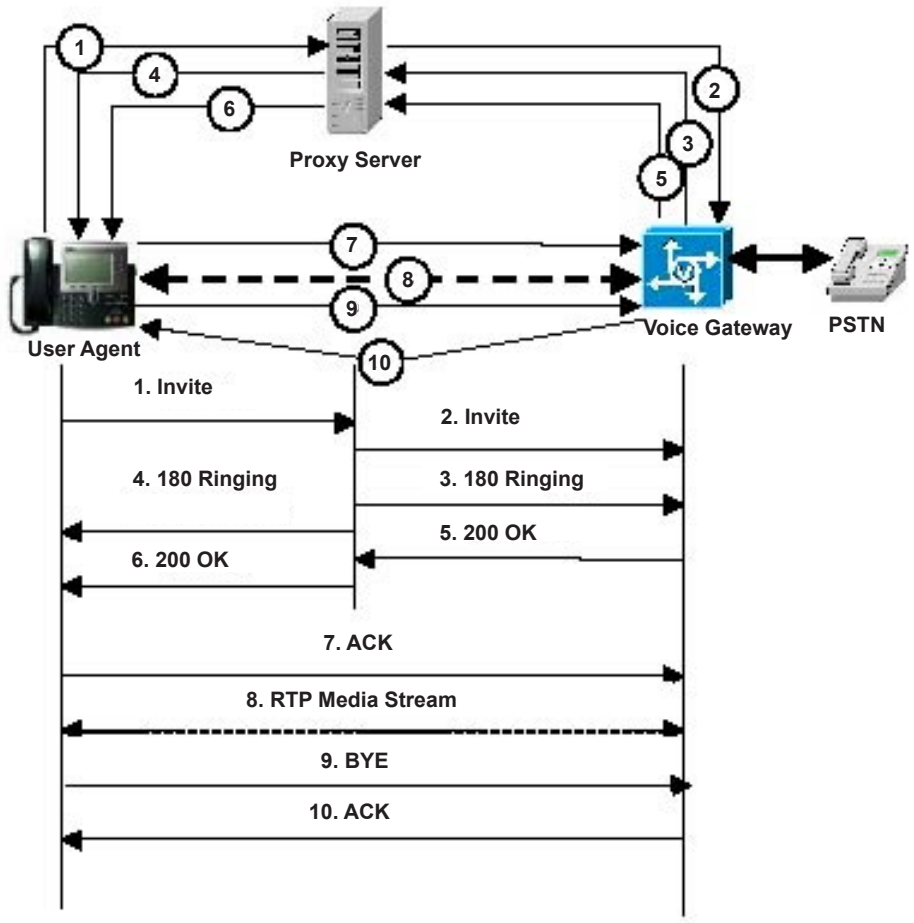


Figure 1: Simple SIP Call

The user agent initiates a call by sending an INVITE message to the proxy server. The INVITE message contains the information necessary for the target endpoint to open a media channel to the user agent. This information includes the IP address and port of the user agent, and the codecs that the user agent supports. The proxy forwards the INVITE message to the voice gateway (when the proxy receives the INVITE message from the user agent, and when the voice gateway receives the INVITE from the proxy, a 100 Trying response is returned, but that is not shown in the call flow diagram). The voice gateway sends out a signal initiating the call to the PSTN, and informs the proxy (which in turn informs the user agent) that it is ringing the destination. When the call is answered, the voice gateway sends a 200 OK back to the proxy. This message contains the gateway's information that the user agent needs in order to set up a media stream back to the gateway. When the user agent receives the 200 OK, it responds directly to

the gateway with an ACK. At that point the call is in session and the two endpoints pass media directly between each other. When either side wants to terminate the call, a BYE is sent from one endpoint to the other, and an ACK is returned.

In this example, once the proxy facilitated the call between the user agent and voice gateway, it dropped out of the call flow, and the two endpoints continued the session management directly between themselves. In some cases, it may be desirable for all session management to flow through the proxy. This may be for security reasons (i.e. the voice gateway is configured to accept messages only from the proxy), or because the proxy needs to be aware of all messages in order to supply certain functionality. SIP defines a “record-route” tag that is inserted in the SIP message by the proxy for such situations. All subsequent messages must then flow through the entity (or entities) addressed in the “record-route” tag(s).

Registration

SIP defines a method for a user agent to inform a server of its location. The user agent “registers” with the registrar of its domain, so that when a SIP message headed for that user reaches the domain, the registrar knows where to forward the message. The registration message contains a “TO” field that defines the SIP address that messages are intended for, and a “CONTACT” field, which holds the forwarding address where the user can currently be reached. The registration is valid for an amount of time defined in the EXPIRES parameter.

The following scenario shows an example of a user agent registering with a registrar, and then being contacted by another user agent via that registrar.

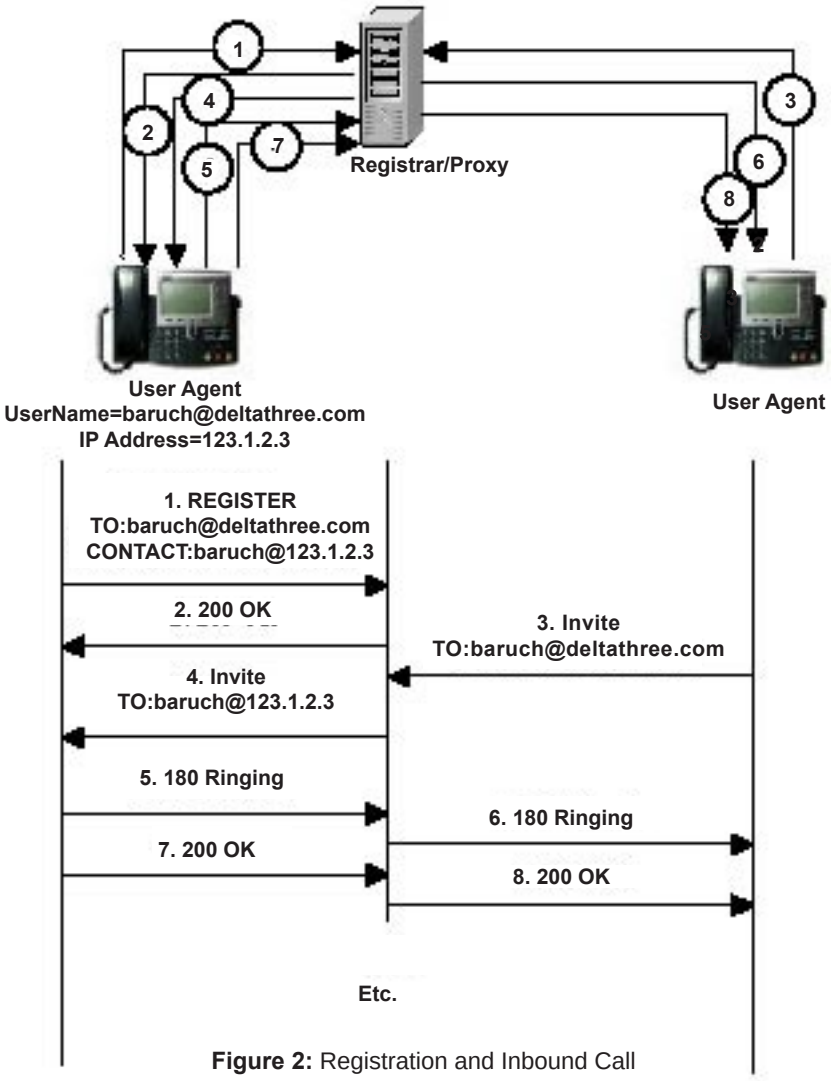


Figure 2: Registration and Inbound Call

The first user agent (*baruch@deltathree.com*) registers with a registrar and sets its contact information to baruch@123.1.2.3. Another user sends an INVITE over the Internet to *baruch@deltathree.com*, and that message is routed to the SIP proxy/registrar for the domain deltathree.com. The proxy/registrar looks up the user *baruch* in its registration database and forwards on the INVITE message to the user agent at IP address 123.1.2.3. From then on, the call flow continues as in the previous section.

Authentication

In order to deny service to any request that does not come from an authorized user, SIP defines an authentication protocol that allows users to identify themselves with a username and password in a secure manner. SIP supports both basic and digest authentication; basic sends the username and password in clear text while digest passes the password in encrypted form. When a user agent attempts to register or sends an INVITE to a proxy, the proxy/registrar responds with a 401 (or 407) response, indicating that authorization is required (if digest authentication is used, that response also contains the information necessary to build a secure transmission which includes the encrypted password). When the user agent receives the response, it will reissue the REGISTER or INVITE, but this time it will include the necessary authentication information (incorporating the security information just received, in the case of digest authentication). The proxy/registrar will test the username and password against its own data, and if they are valid, it will process the request accordingly.

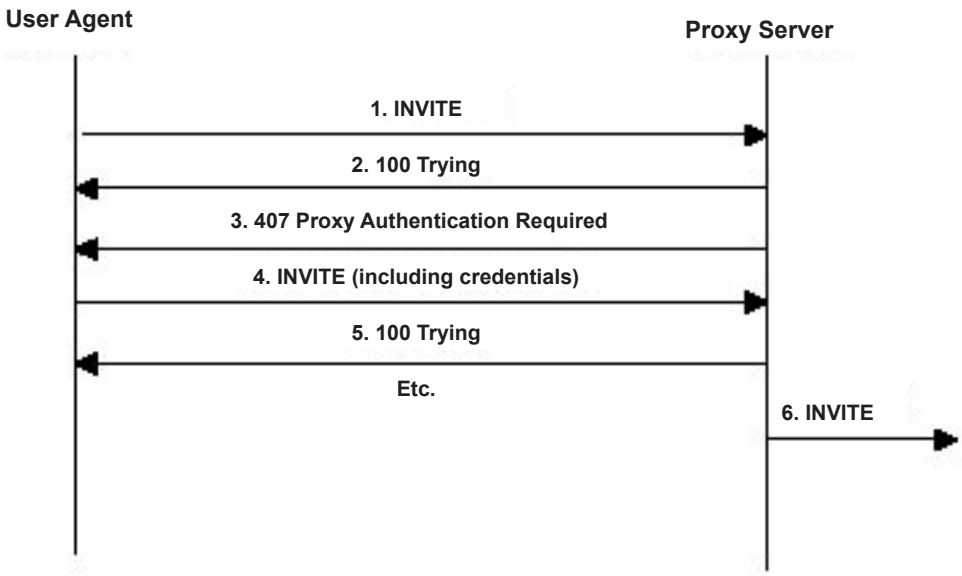


Figure 3: Authentication Flow

Additional SIP Capabilities

SIP was designed to be open and flexible, and as a result it can be used for many diverse applications. One can build on SIP's location and session negotiation capabilities and apply them to various situations. In some cases, extensions have been defined that stretch SIP's functionality even further. Also, SIP is still very much a work in progress, and various and subtle add-ons are being suggested and worked on.

Some additional applications where SIP is being used include:

- Presence – SIP's registration infrastructure provides the foundation for centralized tracking of users' location and status. SIP can be used to build 'buddy' lists and other presence based programs.

- Instant Messaging – IM applications typically send short text messages between parties. SIP can accomplish this easily since it already has a location mechanism and supports text attachments to the SIP message.

- Event notification – SIP defines a Subscribe/Notify message header that is used by a user agent to request that a server or user agent inform it when certain events occur. One example of this might be an IP phone subscribing to a voice mail server and requesting that it be notified when a message comes in. The IP phone would receive a message in the event that mail is received, and the phone might then switch on a "message waiting" indicator to let the user know that the voice mailbox should be checked.

There are numerous other examples of SIP-based applications. Any situation that requires communication over the Internet between two or more endpoints might be implemented using SIP. As SIP matures and is more widely deployed, and as developers become more aware of its capabilities, we can expect to see SIP-based applications becoming ubiquitous.

Prepaid Billing

In February of 2001, I attended a SIP conference where a well-respected expert was discussing SIP-based Internet Telephony. I stood up and asked him how SIP intended to deal with billing, specifically with per minute charges. Since SIP deals mainly with sessions, per session billing is not a problem. But, as an Internet Telephony Service Provider (ITSP) that bridges calls between IP based endpoints and the PSTN, our costs are based on the PSTN billing model (per minute), and so we need to charge our customers accordingly. The speaker explained that within a few years, the legacy telephony system would be completely replaced by VoIP, and just as no one thinks of billing per email, so too telephone calls would be bundled in a flat monthly service charge for IP connectivity.

Needless to say, I was not convinced, and that night, proceeded to write an Internet draft that proposed a method of interfacing SIP calls with RADIUS (a standard protocol used by many billing systems). A few months later, Microsoft came out with their requirements for ITSPs wishing to terminate SIP-based traffic from their Messenger client, and per minute charges were an essential component. As long as telephony costs are linked to usage, Internet Telephony must reflect that in its billing model, and must bill in real time. If not, the potential fraud risk will be enormous, as clever hackers will find ways to take advantage of loopholes, to exploit the reliance on statistical methods (like building a rate plan based on assumed average usage), and to use any delay in updating or processing billing information in order to overuse an account's resources.

Real-Time Billing Considerations

The majority of retail sales for VoIP applications come from PC-to-Phone (a software client running on a PC), or Phone-to-Phone (via Interactive Voice Response (IVR) gateways). Users typically sign up for these services via the Web, or purchase calling cards with account information already provisioned. In either case, the model is normally a pre-paid one, where a defined amount is credited to an account and the balance is updated immediately upon termination of a call. It is essential that debiting the account balance occur as close to real-time as possible, since an account must have the updated balance information available for the follow-on call. Before each call, a user's account balance is queried to make sure that there is enough credit to authorize the call. The balance is translated into duration based on the destination/origination rates for that particular user and call, and a timer is set in order to tear down the call if the user goes beyond the allotted duration available.

The algorithm for processing this type of billing structure is often broken into three steps, Authentication, Authorization, and Accounting (AAA).

Authentication

The first stage involved in processing a call request is to retrieve an account identifier (e.g. username) and password. In IVR systems, these are digits entered from the telephone keypad. In the case of IP-based clients, they must be passed to the server in a secure manner, (in SIP this is accomplished through basic or digest authentication). Authentication means verifying that the account is valid and open for making calls, and that the password matches one stored and associated with this account.

Authorization

The second stage verifies that the user is authorized to make this specific call (after the destination, or B-number, for the call is provided). Given a user's rate plan, account balance, and the explicit rates based on the origination and destination associated with this particular call, the billing server will return a maximum duration available. With this duration value available, the system (trusted user agent or network server) will start a timer and terminate the call if the call goes beyond the allotted duration. In the case of IVR-based calls, the voice gateway will typically have the timer functionality and the ability to close a call. In the case of IP-based clients, call-teardown poses a more complicated problem, since the client usually cannot be trusted to terminate a call when funds run out.

It is imperative that the data available to the billing server for this calculation be current. If there is a delay between the end of one call, and the account balance update, a user placing a follow-on call immediately after a previous one will have more credit available than should be the case. The maximum duration set for the follow-on call will allow the user's balance to run below zero creating a liability in terms of collecting the funds owed.

I would like to stress the critical nature of fraud associated with IP based calling products. On the one hand, the automated, online (and somewhat anonymous) nature of the provisioning process makes these applications particularly vulnerable to hackers and credit-card thieves. On the other hand, the costs incurred by the ITSP are real in that every minute originating from the ITSP's network and terminating on the PSTN will be charged to the ITSP. So there is an immediate liability in terms of real money as soon as an account is opened, and a system that allows any loophole whereby a user can spend money without proper coverage will be exploited to the benefit of rogue IP telephony hackers.

Accounting

The last stages involved in the billing process are the recording of the call details (CDR), the final rating of the call, and updating the account balance. Again, this accounting must be done quickly in order for the follow-on balance to be accurate. To accurately determine the duration of the call, either the call start and stop information must be recorded and then correlated, or else at least one element in the network that can be trusted to know precisely the start and stop of a call must keep state and track the duration.

Centralized Authentication – Digest Authentication to RADIUS Mapping

SIP has no preferred method for implementing AAA functionality or billing. Indeed, since SIP is meant to manage the call session, set up and tear down the call, it could be argued that billing is completely orthogonal to the scope and functionality of SIP. However, even in the earliest drafts, the authors of SIP began to address certain issues relating to AAA – namely authentication. Sessions originating on the Internet, or other IP networks do not have the same trusted stature of a call coming from a dedicated twisted pair of copper wires that are physically attached to a device located at a definite address. In fact, the idea that authentication of the originating user is essential, and the notion that only trusted users will have access to certain services and applica-

tions available through SIP implies either that those services may involve confidential information, or that those services are billable (note that generally, email does not require authentication of the sender, because, the outbound-only nature of email means that the sender is not accessing private information – only the receiving party has that constraint and so must provide a password. Also, email is virtually never billed for on a per-email basis).

Whereas authentication was always considered part of SIP, authorization and accounting were not addressed. Most commercially available proxies have logging functionality that can usually be adapted for the purpose of CDR creation. Reading logs, however, implies a “batch” mindset where records are processed off-line at a later time. They are not well-suited for real-time billing applications. The fundamental requirements for real-time billing as described above are a central billing service where each of the three A’s read from and write to the same data repository, and that information is updated transactively, that is a follow-on call is available only once the previous call’s information is processed and the balance updated. Systems working in this way need to “push” the information to the billing server at the termination of a call, and not rely on writing the call information to some file or queue and waiting for the server to “pull” that information and process it.

Many legacy systems implementing real-time AAA use a protocol called RADIUS (Remote Authentication Dial In User Service) [3][4]. RADIUS is meant to provide the following functionality:

- Passing a user’s password securely, not in clear text.
- Verifying that the RADIUS client is a trusted entity.
- Verifying that the RADIUS server is a trusted entity.
- Passing additional information between the client and server in a secure manner, ensuring that no one in between can tamper with the data.

To accomplish this, RADIUS makes use of a shared secret between the client and server. That shared secret must be specified to each party in a prearranged manner that is outside the scope of the RADIUS protocol.

Note that confidentiality is assured only to the password field, all other communication is guaranteed reliability (i.e. no one can change the data), but theoretically someone sniffing the data going back and forth could read the information.

RADIUS defines (among some other ancillary things) two messages, the authentication message and the accounting message. A client sends an authentication request including the username and password to the server which responds with an accept or reject message. The accounting message has two subtypes; accounting start, sent at the beginning of a call, and accounting stop, sent on termination of a call, which includes all the information needed to create a CDR. All of these messages may also include Vendor Specific Attributes (VSAs), and in that way the client and server can pass information between them in a reliable manner. The RADIUS server will pass the information over to a billing service that reads and writes to a database and processes

the information for the RADIUS server in order to determine how the server should respond to requests and what information it should pass back to the client.

The second A – authorization – requires the destination or B-number of the called party, and based on that information, determines the duration available to the user for this call. This can be implemented through the use of VSAs in either the authentication message or in the accounting start message.

As previously mentioned, in SIP, the username and password are verified using digest authentication (when security is a necessity). However, due to the different encryption and hashing (one way encryption) mechanisms used by digest and by RADIUS, there is no direct way of using the digest information and translating it to RADIUS (the details of this are beyond the scope of this paper, and can be found in [5]). We have proposed a mapping between SIP digest authentication and RADIUS that requires no changes to the SIP client, but necessitates modification of the SIP proxy and RADIUS server (these modifications have already been implemented in commercially available SIP Proxies such as Cisco's CSPS product). In this manner, a SIP proxy can authenticate a user from the data provided through RADIUS to a centralized server. Other SIP network elements may take care of authorization and accounting, again via RADIUS, to that same centralized server. Such a setup will take care of the fundamental requirements for a real-time interface between a SIP network and a billing service.

Call Teardown and CDR Creation

Now that a real-time interface between the SIP network and billing service has been defined, the only thing still required in order to realize a pre-paid model is call teardown when the call reaches the maximum duration allotted. There are two suggested approaches as to where the element that is responsible for call-teardown should be located. Each option has advantages and disadvantages and the ultimate decision will depend on the specific network topography and other considerations.

Control at the edge

The first model places the responsibility for tearing down a call at the edge – in the voice gateway that either terminates the call to the PSTN or originates the call from the PSTN. Many gateways (such as Cisco's voice gateways) have the functionality already in place to communicate via RADIUS, set timers, and teardown a call when the maximum duration is reached. The reason this functionality has already been implemented in the gateways is that they are often used as IVR gateways for pre-paid calling card (phone-to-phone) applications. Very minor modifications are necessary in order to get the same functionality to work from a SIP-based VoIP call, since all the necessary information is already present in the SIP message, including the B-number or destination, and the user information. When the call terminates, the accounting stop message is issued to the RADIUS server by the gateway passing all the CDR information back to the centralized billing service in real-time.

An advantage of keeping this functionality at the edge is that the gateway is one network element that must always, in any case, keep state for the entire length of the call. It will therefore always

be able to know the final duration of the call and can pass that information back to the billing service for processing. Without the duration, the billing process would need to correlate accounting start and accounting stop messages, adding to the processing time and resources required by the billing service.

A second advantage is that this solution scales with the gateways, as the number of ports required within a network grows, the number of gateways increases proportionally and the increased processing power needed for keeping state of all those calls is distributed over the gateways.

The major disadvantage of this scheme is that it requires that gateways be trusted elements not only within the SIP network, but also as billing entities. An ITSP may not be able to live with such a constraint if it wants to leverage least-cost routing to various partner termination (or origination) networks. If the gateway does not belong to the ITSP, then it may not be compatible with the RADIUS protocol implemented by the ITSP, or it may not be trusted to pass secure information across network boundaries. Along the same lines, there can be no billing for calls where both endpoints are untrusted entities, or do not speak RADIUS. Calls from IP devices or software clients to each other, or to services like conference servers, voice mail programs, etc. that do not have built in RADIUS support, and in any event would hardly be considered trusted entities, would not be billable.

Centralized call control

The second model places the responsibility for call-teardown in a centralized network element, the call controller. All SIP messages must pass through this element, which is responsible for authorization of the call (via RADIUS), for setting a timer that fires an event when the maximum duration is reached, and for sending out the SIP BYE messages to both parties when that event fires in order to tear down the call. The call controller would keep state of the call (from INVITE and 200 OK through to the BYE or timer firing) and write the accounting stop to the RADIUS server including the call duration.

The advantage of this scheme is that any SIP call passing through the call controller can be billed regardless of the termination or origination characteristics of the endpoint. The disadvantage is that as a centralized network element, it is a single point of failure for network traffic and must be redundant, highly available and scaleable in order to process and keep state on every call running through the network.

Conclusion

Session Initiation Protocol stands poised as the method of choice for management of media transport sessions between Internet-based endpoints. Its advantages are numerous; it is highly extensible and flexible, easy to program and troubleshoot, and is particularly well-suited for the Internet. As such, it has already been adopted by important bodies and companies as their communication protocol, device manufacturers are building it into their products, and software developers are busy implementing applications and services that interoperate using native SIP.

The challenge for the service provider is to integrate SIP devices and software clients with the services and applications available, and bill for them. SIP by itself does not adequately address the problems and requirements associated with billing, specifically real-time billing in a pre-paid model that fits the particular demands and constraints of Internet users. An interface between SIP and legacy billing services is required. RADIUS offers a good prospect as the protocol for that interface since many billing services already support it, and the required mapping between SIP messages and RADIUS can be defined in a relatively straight-forward manner. Development of specific billing entities that sit within the SIP network, or some modification of existing network elements, may be required in order to achieve full integration between SIP devices and applications, and a real-time, pre-paid billing model.

References

[1] M. Handley et al., “SIP: Session Initiation Protocol,” RFC 2543, March 1999.

[2] H. Schulzrinne and J. Rosenberg, “The Session Initiation Protocol: Internet-Centric Sig naling,” IEEE Communications Magazine, October 2000.

[3] C. Rigney, “Remote Authentication Dial In User Service (RADIUS),” RFC 2865, June 2000.

[4] C. Rigney, “RADIUS Accounting,” RFC 2139, June 2000.

[5] B. Sterman, “Digest Authentication in SIP using RADIUS,” IETF draft, draft-sterman-sip-radius-00, February 2001.