Internet Engineering Task Force SIP WG
Internet Draft J.Rosenberg,D.Drew,H.Schulzrinne
draft-rosenberg-sip-firewalls-00.txt dynamicsoft,Level 3,Columbia U.
February 22, 2000
Expires: July, 2000

Getting SIP through Firewalls and NATs

STATUS OF THIS MEMO

Abstract

This document discusses the interaction of the Session Initiation Protocol (SIP) with with Network Address Translators (NATS) and firewalls. We show the difficulties in SIP traversing these devices, and we compare the solutions that might be used.

1 Introduction

The Session Initiation Protocol (SIP) [1] is a general purpose tool for the initiation, modification, and termination of sessions. As a core part of its functionality, SIP must carry around the ports, IP addresses and domain names needed to describe the sessions it controls. It also causes session traffic to be established (for example, RTP [2] streams with audio and video), often on dynamic UDP ports. As such, there are two issues in getting SIP to traverse NATs

and firewalls. The first is getting SIP itself through, and the second is getting the media sessions it initiates through. The latter is byfar the harder problem.

Section 6.40.1 of RFC 2543 briefly discusses some of these issues, but not in detail. This document serves to fill that void by discussing the problems at length, and by proposing numerous architectural solutions, all of which are already enabled by SIP. In Section 2, we define what we mean by firewalls and NATs. In Section 3, we discuss the two scenarios in which SIP and firewalls may interact - internal service, whereby the private network has SIP deployed, and external service, where it does not. Then, in section 4, we identify the numerous problems in getting SIP to traverse a NAT or firewall. Section 5 discusses the two configurations that can be used to address those problems. Section 6 goes through the detailed SIP processing required in both configurations. Finally, Section 7 discusses what we believe are "non-solutions" to the problem.

2 Definitions

2.1 Firewalls

Our definition of a firewall parallels that defined by N. Freed in [3]. A firewall as a device with two interfaces - one on the "inside" and one on the "outside". Its function is generally to protect devices on the inside from those on the outside, and to sometimes prevent users on the inside from connecting to, or accessing, services on the outside.

Firewalls generally come in two flavors. Protocol-end-point firewalls (also known as application firewalls) act as entities for a particular set of application layer protocols. This means they terminate an application layer protocol on one interface, and re-initiate it on another. Application layer firewalls are generally required to implement some subset of the protocol. Their function in this capacity is to implement some safe subset of the protocol, perform validity checks, and possibly authenticate users at the application layer.

The second flavor of firewall is a packet filter firewall. These firewalls do not attempt to terminate application layer protocols. They operate purely at the IP and UDP/TCP layer. A packet filter firewall applies a set of rules and policies to packets received on both interfaces to selectively discard packets. There is no standard set of rules and policies - they are at the discretion of the firewall administrator. However, we observe that several policies tend to be common:

- The firewall lets no UDP packets in or out. TCP packets are not allowed in unless they are destined for specific servers designated to handle the protocol (for example, the designated http server can receive TCP packets on port 80). TCP packets are allowed out, so that users on the inside can connect to servers on the outside. TCP packets coming in that are associated with a connection opened from inside the firewall are also allowed in.

- Same as the previous, except that only specific TCP port numbers are allowed out; typically this includes SMTP, FTP, telnet and HTTP.

- Same as the previous, except that only specific connections with given TCP port numbers from specific hosts are allowed out. This means that users must connect to HTTP proxies inside of the firewall, and that SMTP is allowed only from designated mail relays, for example.

2.2 NATs

A NAT is a logical function, usually embedded in a border router which straddles a public and private network, that translates IP address information from packets which traverse the boundary. Its main application is to hide internal IP addresses of the private network. This is done to avoid renumbering the private network when providers change, to allow a large address space inside the private network to be mapped to a smaller set of addresses on the outside, or to provide privacy. NATs have become popular tools, especially within home networks and small corporate enterprises which cannot afford to purchase large public address spaces.

NATs are stateful devices. They generally require a table to be established listing the active sessions. For each session, the particular bindings and translations are stored. Sessions are either removed explicitly through packets (a TCP FIN, for example, removes TCP session state), or are timed out after an extended period (as is always the case for UDP services).

Since NATs operate at the IP and transport layers, they fail when application layer protocols include IP addresses and ports within their payloads. In these cases, Application Layer Gateways (ALGs) must be deployed instead. These devices have awareness of the particular application, and can translate addresses within message bodies.

The address mappings provided by NAT can be dynamic or static. In dynamic mappings, the set of available addresses is stored in a

table. As new sessions are established, they are assigned one of these addresses. When the session terminates, the address is returned to the free table. In static mappings, there is a one to one correspondence of internal, private addresses, with external, public ones.

NATs come in a variety of types, with each type providing different services and functions. RFC 2663 [4] overviews these different types. We review them here, briefly, to facilitate subsequent discussion.

The types of NATs are:

Traditional NAT: Also known as outbound NAT, these NATs allow communications that are initiated from the private network to the external network. These devices generally translate the source IP address (and possibly port) of outgoing packets, and the destination address (and possibly port) of packets coming back into the private network for the same session.

Basic NAT: A Basic NAT is a type of traditional NAT. It only translates IP addresses, not ports.

Network Address Port Translator (NAPT): NATPs also translate the source port for outgoing packets and destination port for incoming. This allows multiple sessions to be mapped into a single IP address by using the source port as a session identifier. This type of device is extremely useful for home networks connected to a cable modem or DSL line that provides a single IP address.

Bi-Directional NAT: Also known as two-way NAT, these devices allow hosts in the external network to initiate sessions to hosts on the internal network. This is accomplished by deploying a DNS ALG which creates address bindings when external hosts contact DNS for the address of an internal host.

Twice NAT: Twice NAT is used when the address space within the private network overlaps with the address space of public addresses on the external network. With twice NAT, sessions initiated from inside the private network require translation of both the source and destination IP addresses. They also require a DNS ALG. See RFC 2663 [4] for more details.

A related technology, called Realm Specific IP (RSIP) [5] also allows a private network with private addresses to communicate with external

hosts. Unlike NAT, RSIP requires clients to be aware of the differing address realms internally and externally. Whenever they wish to contact and external host, clients connect to an RSIP server to obtain an address within the external name space.

In the discussions which follow, we do not consider twice NAT or Realm Specific IP.

3 Scenarios with SIP

A SIP network consists of proxies, redirect servers, registration servers and user agents. For the purposes of discussion, we collectively call redirect, registration and proxy servers "SIP servers". The distinction between the three is not particularly relevant for purposes of the firewall discussion. In order to receive calls, a user must generally be a customer of a "SIP Provider"; this SIP provider allows the user to register with a SIP server, and receive incoming calls through the SIP server. Outgoing calls may also be made through this server, but it is possible for a user to make outgoing calls without a SIP provider. A user can even receive calls without a SIP provider, but this requires the calling party to know the IP address or hostname of the called party. This knowledge is unlikely in real deployments.

A SIP provider is not necessarily the same as the network access provider for a particular user. In fact, for residential customers, there is a strong trend towards the separation of access services (provided through traditional ISPs like MCI and AT&T Worldnet), and application services, such as email, web, and instant messaging, provided by application service providers (ASPs) like Yahoo and AOL (of course AOL is also an ISP). VoIP service, enabled by SIP, is another example of an application an ASP might provide.

The problem of getting SIP through firewalls and NATs generally only occurs when the user's access provider is a corporate network. Residential ISPs almost never deploy firewalls, nor do universities [1] which follow, we assume that the access provider is a corporate network.

Given this, there are two configurations of interest when getting SIP through a firewall. The first is when the user wishes to use SIP services from some ASP, independently of whether their corporate network or ISP is providing SIP services. We call this first scenario external service. The second scenario is when the corporate network

---

[1] The authors strongly recommend changing ISPs if your residential ISP has a firewall.

is also acting as a SIP ASP (we call this internal service).

3.1 External Service

The configuration for external service is shown in Figure 1.

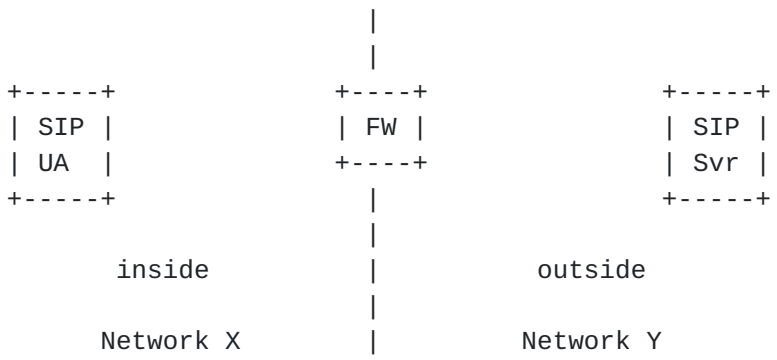


Figure 1

This scenario arises when a user is connected through some network X (usually a corporate intranet), and wishes to use SIP services from some SIP ASP Y, external to X, and X is using a firewall. Generally, firewalls are deployed in corporate enterprises, and enterprise users tend not to make use of external ASPs for services. Rather, these services are available inside of the firewall, so that they can be managed by the enterprise. This configuration therefore occurs in cases where the enterprise has not deployed the service, but employees of the enterprise wish to make use of it through external providers. It also occurs when individual users, for non-business reasons, wish to access service from an outside provider even though it is provided as a service inside the enterprise.

As it turns out, providing SIP services in this scenario, without adding SIP awareness to the firewall, is nearly impossible (without seriously hacking protocols, at least) with any but the most forgiving firewall. The reason is not SIP itself, but the media sessions which are initiated by SIP. Unless the firewall is willing to act as a proxy for external services (see Section 5.1), the firewall is not likely to let the media in or out of the firewall.

This scenario is best solved using a proxy co-located with a firewall, described below in Section 5.1.

3.2 Internal Service

The second configuration occurs when SIP is deployed within the corporate network. In this case, the SIP servers for the user exist on the inside of the firewall, and other network servers belonging to other providers are on the other side. This configuration is shown in Figure 2.

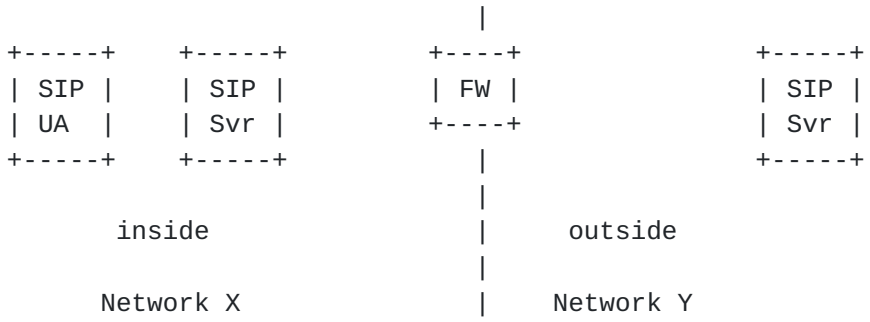


Figure 2

This scenario can be handled with either a proxy co-located with the firewall/NAT, or with a firewall/NAT controlling proxy.

4 Problems in Traversing Firewalls and NATs

Numerous issues need to be considering when using SIP through a firewall or NAT. Both devices come in two general types - network layer or application layer. Network layer devices do not look at the application protocol. Their rules and processing are purely based on IP and/or UDP/TCP processing. Application layer devices operate based on detailed knowledge of the application layer protocol. Many of the issues in getting application layer protocols to traverse NATs are discussed in [6]. This document refers to application layer NAT devices as Application Layer Gateways, or ALGs.

Based on the guidelines in [6], there are numerous problems in getting SIP through a NAT without an ALG, many of which also pertain to firewalls.

These issues are discussed in the subsections which follow.

4.1 Session Bundling

The NAT friendly application design guidelines [6] recommends against protocols which provide session bundling; that is, they use a single session as a control channel for other sessions.

SIP, fundamentally, is a control channel for establishing other sessions (namely, the media sessions). These kinds of protocols (of which FTP is another example) cause problems for NATs, since the addresses for the established sessions are in the body of the application layer messages.

When used with SDP [7], SIP messages carry the IP addresses and ports that will be used for the media sessions. There may be multiple media sessions within a particular SIP call. Since SDP carries IP addresses and not host names, the external UA will send media to an IP address that is not globally routable. Therefore, media will not flow in the direction of external to internal.

A nearly identical problem exists for firewalls. When a user inside the firewall sends media to an address outside the firewall, it will be dropped by the firewall unless a rule is established to allow it to traverse.

Unfortunately, SIP does not mandate that SDP be used exclusively (although all implementations must support it). Future session description formats may be defined.

4.2 IP addresses within Packets

Numerous SIP headers contain fields that can carry either IP addresses or domain names. These can cause problems for both firewalls and NATs.

The Contact header in SIP requests and responses contains a SIP URL. SIP URLs can contain an IP address or a hostname. This address is used for signaling in the reverse direction which is part of the same SIP call. In the case of a NAT, if an IP address is present, the signaling will not be routed correctly from the external network to the private network. If a hostname is present, standard DNS ALGs can provide a translated address through the DNS query, avoiding the need for translation within SIP. If a firewall is in use rather than a NAT, the signaling may not traverse if a rule is not established in the firewall to allow the SIP messages through. In the case of internally provided SIP services, it is easy to configure such a rule. This rule would allow all UDP and TCP traffic on port 5060 to the IP address(es) of the proxies. These proxies form a "buffering zone", performing various checks and validations on the SIP messages,

and acting as a buffer for DoS attacks. However, in the case of external services, the firewall rule cannot include a destination IP address. This significantly worsens the protection provided.

The Record-Route and Route headers in SIP requests have the same issues as the Contact header. When present, they indicate routing instructions for subsequent messaging. Should these include SIP URLs that contain IP addresses from the private network, signaling will not be routed correctly from the external network to the private network in the case of a NAT. In the case of a firewall, rules allowing traffic to be sent to the proxy must be installed.

The Request URI in SIP requests contains a SIP URL. This may contain an IP address. For calls made from the internal network to outside the network, there is no problem, as this address refers to an external host with a public address [2] external network internally, the request URI will usually not contain an IP address (most users do not publish IP addresses within SIP URLs). If an IP address were present, it would have to have been the result of a domain name lookup external to the network. If a NAT were in use, a DNS ALG will allow queries for the IP address of a host within the network to return a globally routable IP address. This implies that special processing, beyond DNS ALGs, is not needed for request URI handling. Similarly, a firewall will need to be configured to allow SIP traffic destined for a proxy internal to the network, in the case of internally provided service.

The Via headers in SIP requests contain IP addresses. These addresses are used to forward responses. Fortunately, SIP supports receiver tagged via fields. If a request arrives from a host, and the source IP address in the packet containing the request does not match the address in the Via field, the proxy tags the Via field with the source IP address. This address is used to send responses. This feature means that the Via field does not need to be translated through NATs. However, the responses are sent to the port in the Via field, *not* the source port of the message. This means that changes to the port numbers, made by NAPTs, will cause responses to be misrouted. To traverse a firewall, it is configured with a rule that allows SIP traffic in destined for the proxy on the inside of the network, as in the cases above.

The Call-ID in a SIP request is required to be globally unique. It is constructed by appending some identifier to the IP address or hostname of the machine where the call originated. This address

---

[2] We do observe that this is a potential problem for twice NAT, however

itself is never needed for message routing. However, if the internal network is using one of the three private address spaces (10/8, 172.16/12 or 192.168/16), it is possible that the Call-ID won't be globally unique. This is because hosts within other private networks might use the same IP addresses. Should two users in two different private networks, both using the same private address, choose the same identifier, and then call the same user, incorrect call handling will take place. The SIP specification does mandate that if the host portion of the Call-ID is not globally routable, the local indentifier must be globally unique, so this should hopefully never happen.

The To and From fields in SIP requests contain SIP URLs, which may contain IP addresses. If a Contact header is not present in the INVITE from a host in the private network, the From field will be used for routing subsequent requests from the called party. If the URL in the From field contains an IP address, these will not be routed properly in the case of NAT. However, most implementations do place a Contact header in the INVITE. Furthermore, since the From and To fields are used for identification of the parties, they generally contain domain names and not IP addresses. Of course, the protocol does not guarantee this.

4.3 Lifetime Issues

When an INVITE message passes through a NAT (and an address binding is established for it) or through a firewall (and rules are created for the media streams), state is established. This state must eventually be cleaned up. For applications that run over TCP, closure of the TCP connection is usually a good indicator of the termination of the application. However, SIP can run over UDP. Thus, determination of when it is safe to destroy state must be done through application level awareness. Unfortunately, an INVITE transaction can last a very long time, as can a call. As a result, it is possible that some state is destroyed before the transaction and/or call actually completes.

In the case of a NAT, the most problematic case for SIP itself (as opposed to the media session) is the IP addresses in the Via field. The received-by parameter in the Via header is populated with the address that will be used to send the response. This address is only valid for the lifetime of the binding. If the binding should expire before the response arrives, the transaction will fail. However, INVITE transactions do not last for more than a minute in the general case. A sufficiently long timeout should ensure this is not a problem.

SIP calls, however, can be for a very long duration (on the order of

hours). Using a set timeout is likely to cause the binding to be lost before the call is complete. If this happens, the name to address bindings obtained from the DNS ALG will be stale. If the DNS ALG returns addresses with very small TTLs, this problem disappears. In that case, the usage of UDP is not problematic; IP address bindings can be re-established for each request (assuming all fields in the SIP message are domain names and not IP addresses).

For the media session, if a NAT times out the address binding, or a firewall removes the rule that allows the media to pass, but the call is not over, the media will not flow and the call will appear as some kind of defect to the user.

SIP can also run over TCP. NATs assume that the IP address bindings have a lifetime equal to that of the TCP connection. In SIP, clients are allowed (and most do) to close the TCP connection to the server once the transaction is complete, even though the call is still in progress. This means that the first problem identified above, whereby the receiver tagged Via fields become invalid if the binding changes before the transaction completes, is avoided with TCP. However, the NAT must be application aware and know not to time out the bindings until the call is actually over.

4.4 Multicast

D.Senie [6] indicates that multicast does not run well through NAT, and an ALG is required. SIP can make use of multicast for both registrations (its most useful application) and INVITE messages. Generally these run on a well known multicast address, 224.0.1.75.

4.5 Security

SIP is secured through both hop-by-hop mechanisms and end-to-end mechanisms. RFC2543 does not mandate a particular hop-by-hop security mechanism; both IPSec and TLS are mentioned. As pointed out in [6], IPSec does not traverse NATs, and TLS is recommended. Therefore, implementations using IPSec for SIP will fail, with or without an ALG. IPSec for SIP through a firewall should work, however, the firewall will not be able to determine the ports used for the media, in order to open up holes in the firewall.

The end-to-end mechanisms in SIP provide both authentication and encryption. Both are problematic for NATs, including ALGs. The authentication mechanisms are not a problem for firewalls, but the end to end encryption is a problem for NAT.

The authentication mechanisms, in particular PGP, sign several fields of the SIP request, including the body. The body, usually SDP,

contains the IP addresses and ports used for the media session. A NAT acting as an ALG will need to change the IP addresses in the SDP. Unfortunately, this will cause the signature to become invalid, and the message to be rejected.

As a fix for this, SIP allows for messages to be resigned. The re-signing party removes the Authorization and Proxy-Authorization headers, resigns them, resulting in new headers, and inserts these into the message. The re-signer then includes their identifier (as a URI) which can be used to obtain the re-signer's certificate. A NAT handling outgoing requests should generally resign the message using the key of the organization itself. This not only solves the authentication problem; it also makes signature verification more realistic. It's much more likely an individual will have access to the certificate for an organization than an individual within that organization.

However, signatures on responses to outgoing requests are very problematic. As responses are forwarded to the internal network by the NAT ALG, the headers may need to be replaced. This would be necessary if the From field or Record-Route in the request was replaced, in which case the original version needs to be put back in the response. Though the NAT ALG could resign the response, this doesn't make sense, since it is not from the same organization as the originator of the response.

End-to-end encryption is much more problematic. It will cause several key headers and the body to be hidden from intermediate systems, including the SIP NAT ALG or firewall. This means the IP addresses cannot be determined or re-written, and holes in firewalls cannot be opened.

> As a result, the use of end-to-end encryption means that SIP will fail through any type of NAT or firewall.

4.6 Conclusion

The conclusion of the analysis of this section is that SIP has several difficulties traversing NATs and firewalls. Most paramount among them is the presence of IP addresses for the media session in the bodies. Additionally, the possibility of IP addresses in the Contact, To, From, Record-Route and Route headers also means an ALG is needed to guarantee operation.

5 Architectural Solutions

As a direct of the discussion in the previous section, there are two

solutions possible for allowing SIP through firewalls and NATs. One is to use an application layer firewall/NAT which understands SIP, and the other is to use a packet filtering firewall/NAT under the control of a proxy.

5.1 Colocation of Proxy and Firewall/NAT

In this solution the firewall/NAT actually is co-located with a SIP proxy. This is shown in Figure 3.

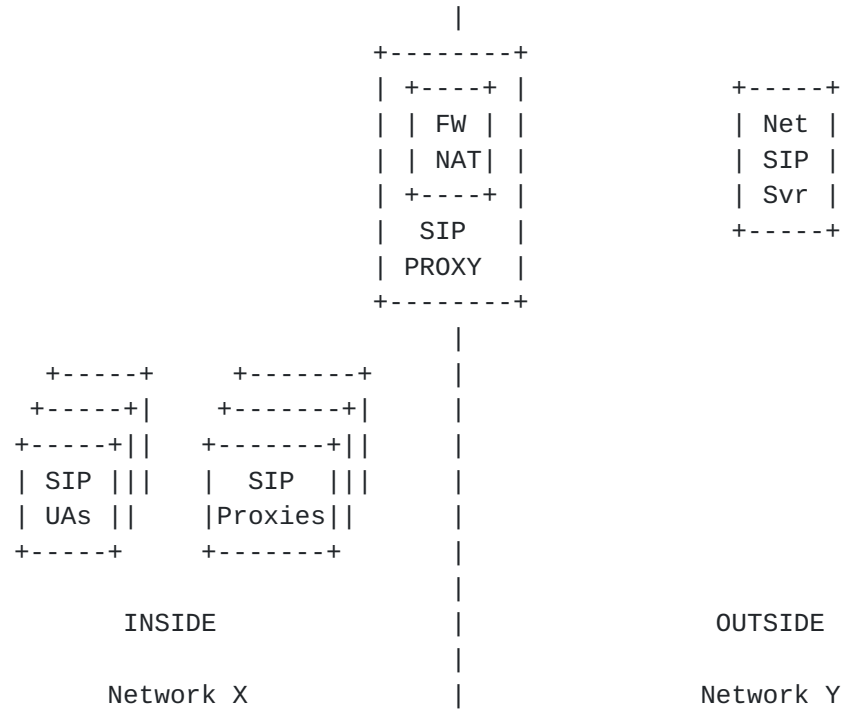


Figure 3

Since the proxy and firewall/NAT are colocated, the proxy can have direct control over the firewall/NAT through some kind of internal API. This configuration is advantageous in that it need not rely on the existence of SIP servers within the network. Although Figure 3 shows proxies inside, the solution allows SIP to work when such proxies do not exist. This makes it the ideal solution for the

external services configuration. A corporate intranet can purchase a SIP enabled firewall, turn on the SIP support, and corporate employees can make use of external SIP services without deploying them internally, while still securing the internal network.

5.2 Firewall/NAT Controlling Proxy

The solution of Section 5.1 has the drawback of burdening the firewall with an application layer protocol. As more applications get deployed, the firewall/NAT needs to become aware of all of them. This eventually may become difficult to manage and difficult to support within an enterprise network.

As an alternative solution, the proxy and firewall/NAT can be separated, but a control protocol or API can be used between them. This protocol would allow the proxy to instruct a firewall to open and close holes for the media stream. It would allow the proxy to query a NAT for address bindings. This allows application layer information to be externalized from the NAT and firewall. This architecture is shown in Figure 4.

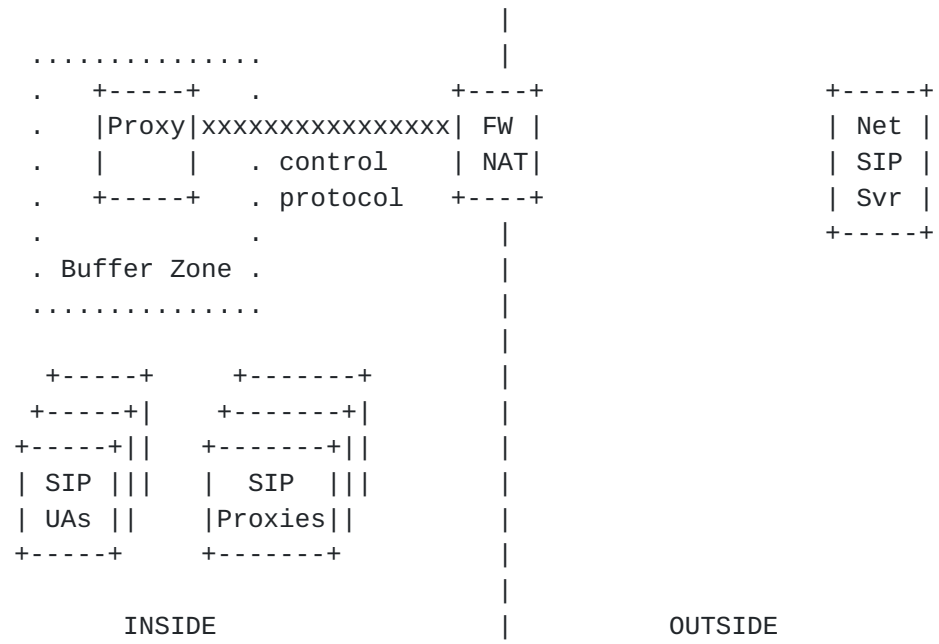


Figure 4

By placing the application layer awareness in the proxy rather than firewall NAT, several advantages are obtained:

- Since the NAT/firewall is focused on a single task - network layer forwarding and filtering, its performance can be optimized. It is not burdened with application layer processing.

- As new applications are deployed, the NAT/firewall does not need to be upgraded. The application servers can control the NAT/firewall as well.

This configuration also requires the firewall to have holes opened permanently for the SIP messaging itself from the proxy out, and from the outside in. This can be accomplished by configuring the proxies with a rule which only allows UDP and TCP on port 5060 to and from the IP address of the proxy (or proxies) inside the firewall. Such a configuration guarantees that the proxy inside the firewall is always the first point of contact for incoming messaging, and the last point of contact for SIP messages before they leave the network. These proxies are effectively part of a "Buffer Zone (BZ)" which exists on the inside of the network, granting access to users and services within.

In the case of a NAT, the proxy in the BZ will need to reachable from outside. This is accomplished by using a DNS ALG that gives out globally routable IP addresses for the proxy when queries for its domain name are given. It is desirable for these addresses to be based on long lived bindings.

The usage of a SIP BZ offers numerous advantages. First, it is a protection against DoS attacks involving flooding of messages. The BZ ensures that all SIP messages hit specially engineered proxy servers, rather than any machine within the network. These proxies can be configured with load checkers which cause error handling to take place when call volumes to any particular host within the network become too high. This error handling might involve rejecting all requests using a stateless mode. This prevents build up of state within the proxy and also protects the internal network.

An additional advantage of the BZ is protection against spam. Certain URLs, known to be from malicious callers, can be rejected outright without bothering internal users at all.

Finally, the BZ allows for message validation. Overly long requests can be rejected. Messages with mailicous Java applets can be detected and rejected. Poorly formed requests can be rejected, particularly ones with very long values for certain fields.

Clearly, this solution only applies when SIP servers are deployed within the corporate network.

Several possibilities exist for the control protocol between the proxies and a firewall. One is SOCKS, specified in RFC 1928 [8] which is engineered for exactly this purpose. We know of no equivalent for NAT control, however.

An additional level of scalability can be achieved by introducing an additional server, which we call a session manager. This device actually stores the call state and other information needed by the proxy servers. In this way, numerous proxy servers can be deployed rather than one. They access the session manager when they need access to call state. The precise way in which this is done requires further investigation.

6 Proxy/Firewall/NAT Operation

Independently of whether the proxy and firewall/NAT are co-resident, there are specific goals to be achieved by the system:

1. Ensure that only authorized users inside the network may make outgoing calls.

2. Ensure that legitimate users outside the network may make calls into the network.

3. Ensure that calls make from outside the network to inside are not malicious or harmful to systems inside the network.

4. Ensure that any bodies carried in SIP messages (such as Java applets or Word documents) are virus free and not harmful to the system.

5. Ensure that media streams for the call are allowed to flow in and out of the network for the duration of the call only.

6. Ensure that users cannot be contacted without having SIP requests routed through the firewall.

The operational procedures required to meet these objectives are outlined in the subsections below.

6.1 Outgoing Calls

When a user makes a call, the proxy acts as a local outbound proxy. This can be accomplished by explicitly or automatically configuring

(through DHCP [9], SLP [10] or ACAP [11]) the local outbound proxy in user agents within the enterprise. This only works when SIP services are provided internally. Alternatively, when SIP services are external, the firewall/proxy can intercept SIP requests not explicitly destined for the firewall/proxy. Doing so requires care. The request needs to still be delivered to the server the user agent sent the request to. To accomplish this, the proxy/firewall notes the destination IP address in the request when the request is captured. When the request is forwarded, rather than determining the next hop server by looking up the host name in the request URI (the normal procedure if the request was actually sent directly to the proxy/firewall), the request is sent to the IP address that was noted previously.

When an INVITE is received by the proxy from a host within the private network, and SIP service exists internally, it authenticates the caller by sending a 407 response. The UAC resubmits the request. If the credentials are valid, the firewall/proxy can check its policy database to determine if the user is authorized to make outgoing calls, and if so, forward the request based on normal SIP procedures. If SIP services are external, no authentication is performed; the request is simply forwarded by the proxy/firewall. In either case, the proxy also adds the Record-Route header. The proxy also remembers the SDP in the INVITE, and extracts the port numbers and IP addresses for each of the media streams.

If NAT is in use, additional operations are performed by the proxy. The IP addresses and ports in the SDP are removed. A binding is created, mapping these address/ports to globally routed ones. This binding is created locally if the NAT and proxy are co-resident. Otherwise, some kind of query is needed to allow the proxy to ask the NAT for a binding. Assuming this is done, the globally routed addresses are inserted into the SDP. This may require the request to be resigned. If the Contact, From, or Record-Route headers contain IP addresses, bindings are created for those addresses as well (these may end up using the same bindings as the IP addresses in the SDP, but not necessarily. As an example, for a megaco [12] decomposed gateway, they will not), and the globally routed ones are placed into the message. If these fields contain hostnames, bindings of those hostnames to globally routed addresses must be done, and a DNS ALG configured to return those addresses when queries are made. The proxy inserts a Via header containing a globally routable IP address and port. If the From, Contact or Record-Route headers were replaced, these must be stored by the proxy. They must be reinserted into the response.

When a response to the INVITE arrives from the external network, it goes through the firewall or NAT (assuming the previous steps have

been followed) and is examined by the proxy. If it is a 200 OK with SDP, the proxy analyzes the SDP, comparing with the one from the INVITE. For all media streams which were accepted by the UAS (the connection address is nonzero), the proxy notes the IP address and port indicated in the SDP in both the INVITE and 200 OK. The response is then forwarded.

The proxy also opens holes in the firewall, if one is in use, to let the media traffic in and out on the ports and addresses obtained from the SDP in the INVITE and 200 OK. In particular, if the INVITE contained address A and port B for some media stream, and the 200 OK contained address X and port Y for the same media stream, the proxy/firewall allows UDP destined for address X and port Y from the inside to the outside of the firewall. Similarly, it allows UDP traffic from the outside to the inside if it is destined for address A and port B. The firewall may further restrict the source addresses; for example, allowing UDP traffic from the outside to the inside if its destined for address A and port B and is from address X. However, this assumes that the called party is using the same machine and interface to both send and receive media. This assumption is not valid for multi-homed hosts, or for multimedia systems with different components running on different hosts. Recent discussions on the SIP mailing list have proposed adding a source address to SDP in order to construct more restrictive firewall rules.

In the case of a NAT, the original From, Contact and Record-Route headers are placed back into the response (note the Record-Route headers placed back will only replace a subset of those in the response). Nothing is done to the SDP for NAT. Replacing headers will invalidate signatures and the Authorization header must be removed, if present.

Recent extensions to SIP allow for "early media" to be opened by including SDP in provisional responses [13] which are sent reliably [14]. As a result, the proxy will also need to look for these responses. The mechanism used to transmit these reliably is based on a new PRACK request. This request can also contain SDP. The firewalls will also need to look for these messages, extract the ports and IP addresses, and open holes based on them or create bindings for them, just as was done for the INVITE. PRACK messages are forwarded based on Record-Route headers returned in provisional responses; this means that the proxy will receive the PRACK messages so long as it inserted the Record-Route in the outgoing INVITE request (which it must do). The proxy can use proxy authentication to verify the authenticity of the PRACK messages as well, if SIP services exist internally.

As a final complication, SIP allows for the INVITE to contain no media information. Rather, the 200 OK contains the receive

capabilities of the called party, and the ACK contains a trimmed subset, representing the capabilities and accepted media streams of the calling party. This requires the proxy to open the holes for media (or create bindings for them) on receipt of the ACK, rather than on receipt of the 200 OK as discussed above.

6.2 Incoming Calls

The initial processing for incoming calls (from the external network to the internal network) depends on whether SIP services exist internally or not.

6.2.1 Incoming Calls for Internal Services

The scenario for incoming calls is largely the reverse of the scenario for outgoing calls. The name space and DNS records must be configured so that all incoming requests for users within the private network arrive at the proxy. If the company is named foo.com, this implies that the SIP URLs published externally for employees of foo.com should be of the form sip:user@foo.com. DNS records must be configured so that a lookup of foo.com (using SRV and/or A records) results in the address of the firewall proxy (or proxies, when there are more than one to support load balancing and backups). SIP URLs are "published externally" through their placement on business cards and personal communications; managing this is outside the scope of this document. However, SIP URL's are also effectively "published" automatically through SIP REGISTER messages. Therefore, users must be prevented from sending external registrations to other servers listing Contact addresses that do not correspond to the address of the proxy/firewall. This is discussed in more detail in Section 6.7.

Assuming the naming and records have been managed properly, an INVITE arrives at the proxy when a call is to be set up to a user inside the network. In the case of NAT, the NAT will have done nothing but rewrite the destination IP address of the packet to the private address of the proxy. All of the SIP level details regarding NAT processing are done in the proxy. Proxy authentication is probably not viable for incoming calls. This is because the domain of the caller is not the same as the domain of the called party. There is unlike to exist any kind of shared secrets between a server in one domain and a user in another. Widespread deployment of a PKI will enable proxy authentication for incoming calls, but it does not exist at the moment.

6.2.2 Incoming Calls for External SIP Service

The scenario for incoming calls is very similar to outgoing calls. The external SIP server will receive requests destined for the user

inside the firewall/NAT. It sends those requests to the user directly. These requests are intercepted by the firewall/NAT, which passes them to the proxy component.

6.2.3 Finishing Incoming Call Setup

The proxy can validate the message, and make sure it is not malicious in nature (for example, ensure it has bodies that are not too long). Once validated, it notes the port numbers and IP addresses in the SDP in the INVITE, but does not open holes in the firewall at this time, nor are address bindings created for the media streams in the case of a NAT. The proxy adds the Record-Route header, which must contain a globally routable IP address for the proxy. Location services are invoked (possibly resulting in next hops with private IP addresses). A NAT will not need to modify any of the fields of the message, as they should all contain globally routable addresses, including the To and Request-URI. The request is then forwarded.

When a 200 OK response (or any provisional response with SDP) arrives from inside, the proxy examines the SDP and notes the IP addresses and ports for each media session. If the 200 OK is signed, the proxy/firewall can verify the signature as an additional security mechanism.

In the case of a firewall, the 200 OK is forwarded as is normally done. At this point, the proxy opens up holes in the firewall for traffic to and from the ports and addresses in the SDP to enable media in both directions.

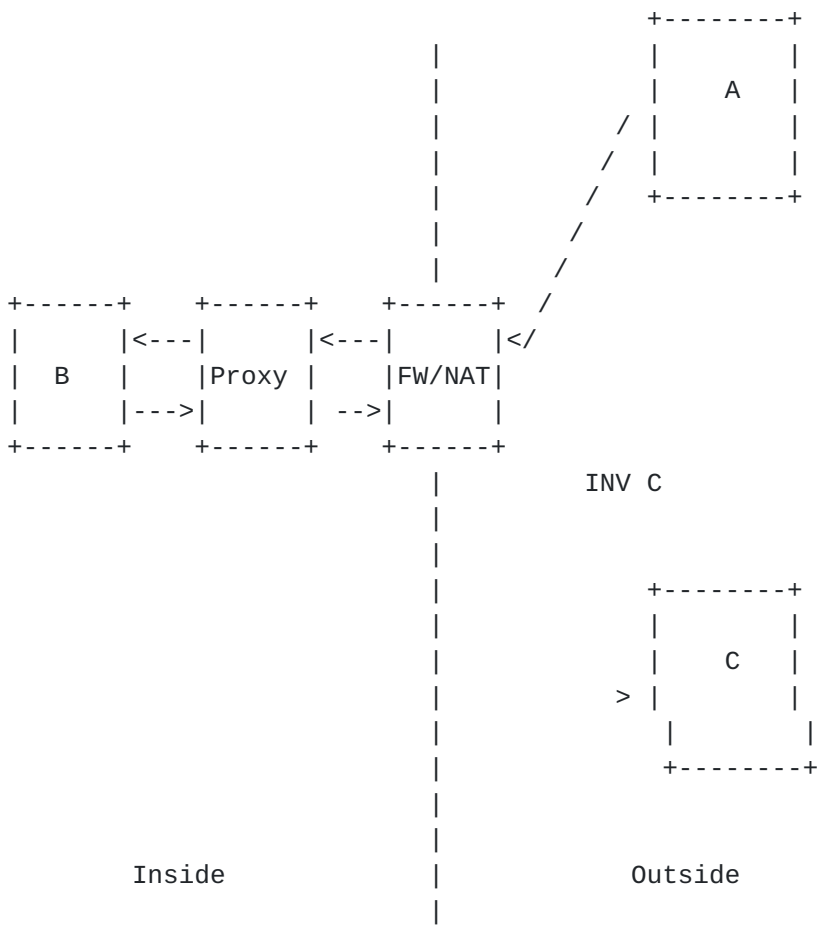
In the case of NAT, the proxy requests a binding for the private addresses in the SDP in the response. The globally routable addresses obtained from the binding request are placed into the SDP. Note that it is OK for the response to contain Contact or Record-Route headers with private addresses or domain names, so long as the proxy itself is record-routing. Thats because these addresses or names will only be needed by the proxy or other servers from within the private network.

When the ACK arrives for the 200 OK, it will pass through the firewall/NAT, and arrive at the proxy. If this ACK contains SDP, the proxy should ensure that the IP address and port have not changed from the INVITE. If they have, new holes in the firewall need to be opened. No special processing is required for the NAT. The ACK is then forwarded.

6.3 Forwarded calls

An interesting case, depicted in Figure 5, arises when a user outside

the network (user A), calls a user inside the network (user B), but this call is forwarded to a user outside the network (user C). When the INVITE from A arrives at the proxy, the procedures for incoming calls apply. However, when the INVITE hits the forwarding logic for user B (present in either B's software or in a proxy server within the network) B, and is forwarded out to user C, hitting the firewall/proxy again [3] , different rules apply than the ones outlined in Section 6.1 for outgoing calls. Although this case appears like an outgoing call (INVITE message from within the network to outside), it is not. The difference is that the From field does not correspond to a user within the corporate network.



---

[3] The astute reader will note that this case appears to be a loop according to RFC2543, although it really isn't. This problem has been identified and a fix has been drafted for the next version of SIP

Figure 5

Specifically, unlike outgoing calls, proxy authentication should not take place for internal service. The call originator is not inside the network. In addition, the proxy does not need to record route, or even be stateful. No holes in the firewall need to be opened for the call, since the media will flow directly between user A and user C (assuming the call is accepted), neither of which is within the enterprise. Neither do address bindings need to be established. It is our recommendation that proxies act statelessly in this configuration.

6.4 Termination

A call is terminated with a BYE message. If the BYE comes from the private network, the proxy can use proxy authentication to authenticate the originator of the request if the service is internal. However, the holes in the firewall are not closed yet, and NAT address bindings are not released. If the original INVITE was generated by a host within the private network, and the From field was replaced as it contained a local address, the global address must be replaced in the BYE. This is identical to the procedure for the INVITE itself, and applies to re-INVITES too. When a 200 OK arrives, the proxy closes the holes in the firewall and releases any address bindings in use by the NAT.

If the BYE comes from the external network, the NAT must replace the To field if the From field was modified in the original INVITE from the private network. When the 200 OK arrives for the BYE, the proxy can verify the signature on the 200 OK (assuming mutual authentication was used).

It is preferable to close the holes in the firewall and release address bindings on the response to the BYE, rather than the BYE itself. In unusual circumstances, the BYE might be rejected and the call remains established. The 200 OK confirms termination of the call.

6.5 Re-INVITEs

SIP allows parameters of the session to be changed through re-INVITEs. These re-INVITEs can add new media sessions to the call, in which case new holes need to be opened in the firewall, and new address bindings may need to be created. This process follows the procedure for original call setup. Similarly, re-INVITEs can remove

media sessions from the call, meaning the holes in the firewall need to be closed and bindings released.

The procedure for opening or closing holes or establishing bindings for re-INVITEs is identical to opening holes for new INVITEs. In fact, since SIP INVITE messages carry the entire session state, the proxy can simply close all the holes associated with the original INVITE, and re-open ones for this INVITE as if it were the first. For NATs, this will not work. If a media stream is removed as a result of a re-INVITE, that address binding, and that one alone, is released.

6.6 Session Timer

As described above, the proxy relies on the SIP BYE message to know when it is safe to close a hole in the firewall or release an address binding in the NAT. However, relying on these messages completely can be problematic for several reasons:

- End system crashes may result in calls ending without BYE being sent.
- Malicious users may never send BYE in order to attack a proxy.
- Poor implementations may not process Record-Route and thus the proxy/firewall may never see a BYE.
- Network failures may cause a BYE that is sent to never reach the proxy.

The solution to this is for the proxy to make use of the session timer extension for SIP [15]. This will allow the proxy to receive keepalives for the call. If the keepalives stop arriving, the proxy can close the holes in the firewall or release the NAT address bindings.

6.7 Registrations

Special consideration must be given to registrations going into, and leaving, the network in the case of internal service. There are legitimate reasons for both to occur. For external service, REGISTER messages are intercepted and then forwarded towards the final destination without any local processing (except to determine that this is a REGISTER message).

In the case of a NAT, note that address bindings are never created upon receipt of registrations. Rather, they are triggered when an INVITE arrives that makes use of that registration.

6.7.1 Outgoing Registrations

Registrations leave the network (i.e., flow from inside to outside) when a user on the inside registers with some external server. A legitimate application of this is when an employee is travelling on business, and has instructed their SIP server within the corporate network to forward all calls to the main server for the facility the employee is visiting (say, bar.com). This registration does not leave the network. However, in order to receive calls at bar.com, the employee may need to instruct bar.com to forward calls to the specific building he is visiting, bldg12.bar.com. This will require a registration sent to the server at bar.com, indicating bldg12.bar.com in the registration. This REGISTER message is sent externally.

An additional case where registrations may leave the network is if a customer has an external SIP forwarding service, perhaps from an organization like acm.org, and they wish to have all calls for them forwarded to work. This requires a registration to be sent from inside the firewall, to the acm.org server, listing their work URL, sip:user@foo.com, in the Contact header.

In all cases, the registrations must be such that calls forwarded to the private will always reach the internal proxy that is controlling the firewall or NAT. This results in the following rule:

> Firewall and NAT controlling proxies should allow SIP registration messages to leave the enterprise. However, if the domain in any of the Contacts is within the corporate network, but the domain of the To field is not, the URLs in the Contact headers MUST contain the firewall/proxy's public IP address or domain name in the host part. If the domain in the To field is within the corporate network, any address, including private IP addresses, are allowed in the Contact headers. Registrations not conforming to this should be rejected with a 600 class response.

Like INVITE messages, it is also recommended that proxies perform proxy authentication on outgoing registrations.

6.7.2 Incoming Registrations

Incoming registrations allow a user outside the network to establish forwarding state within a proxy on the inside of the network. The scenario arises when an employee of the company is travelling, and is now logged in through some external network. They wish to have calls for them forwarded from work to their new location. This may require a registration to be sent from outside the network, to a registration

server inside the network. We say "may", since if the user is connected externally through a VPN service, the registrations will not pass through the firewall, and will be effectively the same as purely internal registrations.

As such, if a corporate intranet has VPN services deployed, and roaming users make use of these services, it is recommended that firewalls disable registrations coming in to the network from the outside. This is accomplished by rejecting, with a 600 response, all registrations which arrive from the external interface.

If VPN services are not enabled, registrations should be forwarded or processed when arriving externally if:

1. The registrations are authenticated, and come from a known user of the network,

2. The To field in the registrations corresponds to a valid URL within the corporate enterprise

Registrations not meeting these criteria should be rejected.

6.8 Call Cancellation

When a call is cancelled, the proxy forwards the CANCEL according to the rules in RFC 2543. The only additional operation to consider is that a provisional response may have been sent for this call, and this provisional response may have opened up holes in the firewall for early media, or may have created address bindings for them. In this case, a cancellation may require the holes to be closed and bindings released for the early media. Before releasing the firewall/NAT resources, the proxy should wait for a certain period for a final 200 OK response to the INVITE which may have crossed the CANCEL on the wire. If none comes, or if a 487 or other non-200 response arrives instead, the holes in the firewall should be closed.

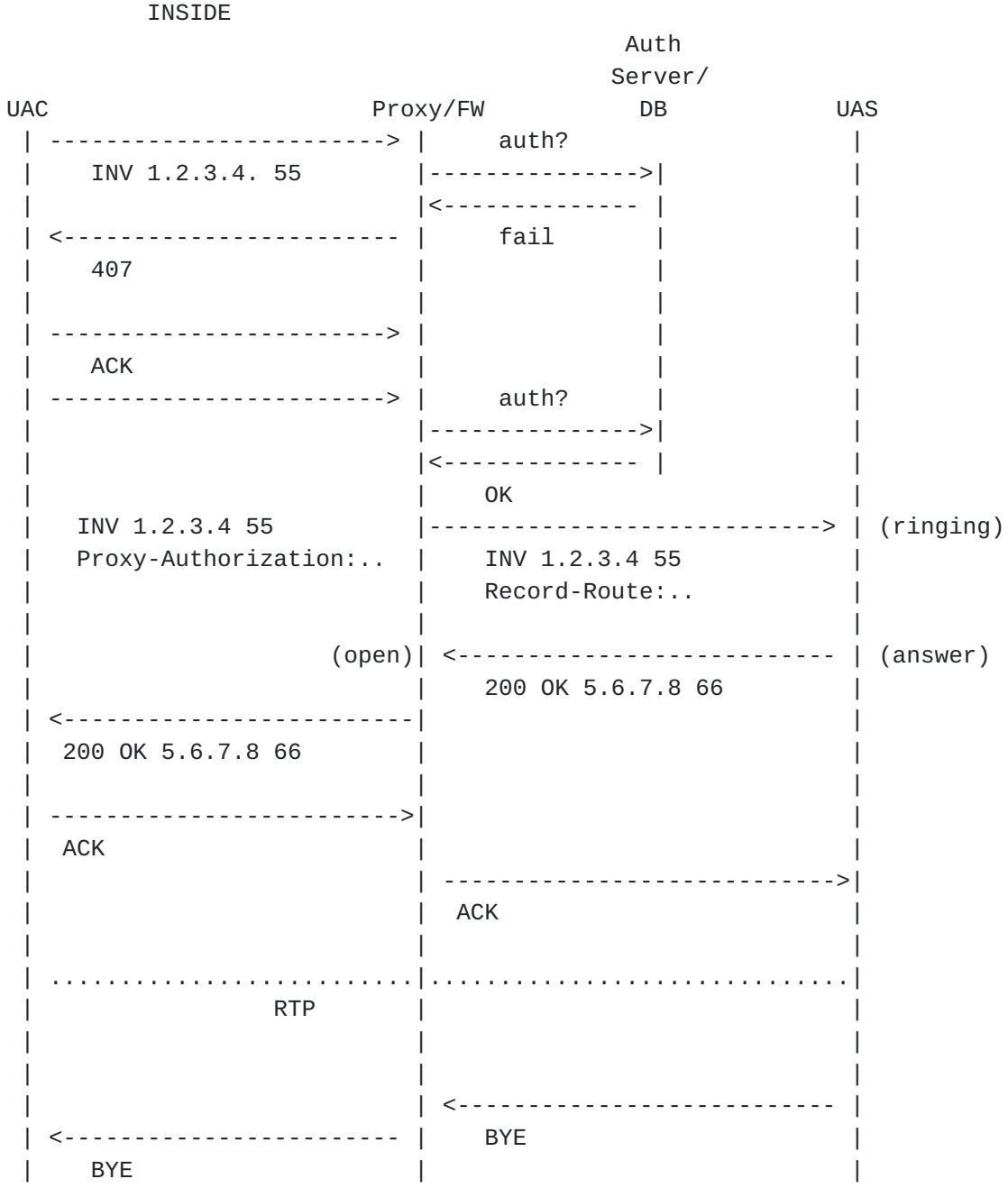
6.9 Additional Messaging

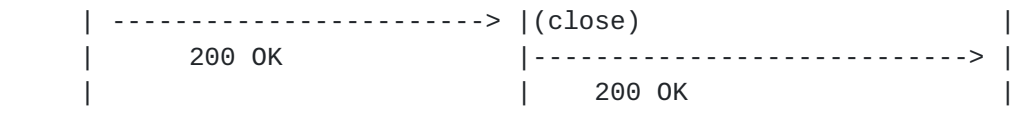
SIP defines additional requests, in particular OPTIONS, which do not come into play in the sections above. Extensions have been developed which specify new requests, such as INFO [16]. Since these requests don't alter call or session state, they should be forwarded normally, possibly after proxy authentication.

6.10 Call Flow: Outgoing call

The call flow in Figure 6 shows an example of an outgoing call setup through a proxy controlling firewall, in the case of a private

network with internal SIP service. In the diagram, "INV A B" means an INVITE message with SDP whose IP address in the c line is A, and whose port in a single m line is B. Similarly, "200 OK A B" means a 200 OK response with SDP whose IP address in the c line is A, and whose port in a single m line is B. "auth?" is an authorization request to a local authentorization server (such as a RADIUS [17]), or a query to a database server, such as SQL or LDAP.





open: Allow UDP, port 55, destination IP 1.2.3.4 into firewall from outside. Allow UDP, port 66, destination IP 5.6.7.8, out of firewall from inside.

close: Do not allow UDP, port 55, destination IP 1.2.3.4 into firewall from outside. Do not allow UDP, port 66, destination IP 5.6.7.8, out of firewall from inside.

Figure 6

7 Non-Solutions

Two alternate solutions have been proposed for allowing SIP to get through firewalls. Both are primarily aimed at solving the external services configuration. We do not consider these to be viable long term solutions for the problem. They are:

1. Run SIP over port 80, which is used by HTTP [18]. Since SIP is "close" to HTTP, this may fool firewalls into letting SIP requests out of the firewall.

2. Have the user agents open a TCP connection out through the firewall to its SIP server. Call invitations for the user are sent over the open TCP connection. Since many firewalls allow outgoing TCP connections, SIP messaging can flow.

The first of these is explicitly frowned upon administrators learn that people are muxing other applications on port 80, and then additional capabilities in firewalls are turned on to block messages based on their content. Furthermore, it is fundamentally a means to bypass administrator policy. It is our belief that the approach for getting SIP through firewalls is to do so cooperatively, where the firewall administrator knows about SIP messaging and explicitly allows it. This is the only solution which can persist long term. Furthermore, although the first solution may succeed in getting SIP out through the firewall, SIP messages won't be forwarded externally into the private network (most firewalls won't let HTTP requests on port 80 into the network to arbitrary hosts). Furthermore, since the media is sent on dynamic UDP ports, the media won't flow through the firewall in all likelihood.

The second solution is similar in concept to the first. It is less antagonistic towards firewall administrators, though. Rather than fooling the administrator, it attempts to fit the service into one of the allowed configurations. It improves upon the first by allowing incoming requests. However, the same problem exists with media streams. Some kind of TCP to UDP translator may be needed in this case.

8 To Do

1. More detail on NAT operation, particularly creation of bindings and detailed message flows.

2. Mention rejecting requests by the proxy that don't contain SDP, as a way to avoid problems associated with chaning the session description format.

9 Author's Addresses

Jonathan Rosenberg
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
email: jdrosen@dynamicsoft.com

Dale Drew
Director of Security Engineering
Level(3) Communications
1450 Infinite Drive
Louisville, Colorado, 80027
email: dale.drew@level3.com

Henning Schulzrinne
Columbia University
M/S 0401
1214 Amsterdam Ave.
New York, NY 10027-7003
email: schulzrinne@cs.columbia.edu

10 Bibliography

[1] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP:

session initiation protocol," Request for Comments (Proposed Standard) 2543, Internet Engineering Task Force, Mar. 1999.

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for real-time applications," Request for Comments (Proposed Standard) 1889, Internet Engineering Task Force, Jan. 1996.

[3] N. Freed, "Behavior of and requirements for internet firewalls," Internet Draft, Internet Engineering Task Force, June 1999. Work in progress.

[4] P. Srisuresh and M. Holdrege, "IP network address translator (NAT) terminology and considerations," Request for Comments (Informational) 2663, Internet Engineering Task Force, Aug. 1999.

[5] M. Borella, J. Lo, D. Grabelsky, and G. Montenegro, "Realm specific IP: a framework," Internet Draft, Internet Engineering Task Force, Dec. 1999. Work in progress.

[6] D. Senie, "NAT friendly application design guidelines," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[7] M. Handley and V. Jacobson, "SDP: session description protocol," Request for Comments (Proposed Standard) 2327, Internet Engineering Task Force, Apr. 1998.

[8] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones, "SOCKS protocol version 5," Request for Comments (Proposed Standard) 1928, Internet Engineering Task Force, Apr. 1996.

[9] G. Nair and H. Schulzrinne, "DHCP option for SIP servers," Internet Draft, Internet Engineering Task Force, Feb. 2000. Work in progress.

[10] J. Kempf and J. Rosenberg, "Finding a SIP server with SLP," Internet Draft, Internet Engineering Task Force, Feb. 2000. Work in progress.

[11] C. Newman and J. G. Myers, "ACAP -- application configuration access protocol," Request for Comments (Proposed Standard) 2244, Internet Engineering Task Force, Nov. 1997.

[12] F. Cuervo, C. Huitema, K. Kelly, B. Rosen, P. Sijben, and E. Zimmerer, "MEGACO protocol proposal," Internet Draft, Internet Engineering Task Force, Mar. 1999. Work in progress.

[13] S. Donovan, H. Schulzrinne, J. Rosenberg, M. Cannon, and A. Roach, "SIP 183 session progress message," Internet Draft, Internet

Engineering Task Force, Oct. 1999. Work in progress.

[14] J. Rosenberg and H. Schulzrinne, "Reliability of provisional responses in SIP," Internet Draft, Internet Engineering Task Force, Jan. 2000. Work in progress.

[15] S. Donovan, "SIP session timer," Internet Draft, Internet Engineering Task Force, Oct. 1999. Work in progress.

[16] S. Donovan, "The SIP INFO method," Internet Draft, Internet Engineering Task Force, Jan. 2000. Work in progress.

[17] C. Rigney, A. Rubens, W. Simpson, and S. Willens, "Remote authentication dial in user service (RADIUS)," Request for Comments (Proposed Standard) 2138, Internet Engineering Task Force, Apr. 1997.

[18] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "Hypertext transfer protocol -- HTTP/1.1," Request for Comments (Draft Standard) 2616, Internet Engineering Task Force, June 1999.