Network Working Group
Internet-Draft
Expires: August 14, 2005

J. Rosenberg
IAB

February 13, 2005

Considerations for Selection of Techniques for NAT Traversal
draft-iab-nat-traversal-considerations-00

Status of this Memo

Copyright Notice

Abstract

There are many protocols designed and deployed on the Internet today which do not naturally traverse Network Address Translators (NAT). In order to allow these protocols to work in the presence of NAT, additional logic needs to be added to the network. This logic modifies the behavior of the protocol in some way. There are choices where this logic can be placed in the network. It can reside in the NATs themselves, transparently altering the protocol; when this occurs, it is called an Application Layer Gateway (ALG). It can

reside in server components, hiding the changes from NATs and clients alike, it can reside in the clients, or it can reside in a combination thereof. The choice of the placement of this logic typically has implications on many aspects of the protocol, including security, deployability, manageability and availability. This document provides a set of considerations that should be taken into account by protocol and network designers when making this choice.

Table of Contents

1. Introduction

A Network Address Translator (NAT) is defined in RFC 2663 [1] as "a method by which IP addresses are mapped from one address realm to another, providing transparent routing to end hosts." NAT and its many variations have seen widespread deployment over the past few years. However, many protocols were designed and deployed before the widespread deployment of NAT. Some of these protocols cease to work correctly in the presence of NAT. These include the File Transfer Protocol (FTP), the Domain Name System (DNS), the Session Initiation Protocol (SIP) [2], and the Real Time Streaming Protocol (RTSP) [3] amongst others.

In order for these protocols to operate on the network in the face of NAT, numerous techniques have been developed, many of which are protocol specific. These include deploying Application Layer Gateways (ALGs), upgraded clients or servers, or one of the many Unilateral Self Address Fixing (UNSAF) techniques [4], such as the Simple Traversal of UDP through NAT (STUN) [5], Traversal Using Relay NAT (TURN) [6], Teredo [7] and Interactive Connectivity Establishment (ICE) [8]. As such, network designers and operators are faced with a choice of techniques.

The right choice of technique depends on a number of architectural considerations. The purpose of this specification is to provide guidance to network designers, planners and operators on the selection of general technique for NAT traversal. It is not a detailed comparison of STUN, TURN, Teredo, and ICE, but focuses on the architectural issues around the choice between modified application components, ALGs, and UNSAF techniques. Many of the considerations here also apply to the design of traversal techniques for new protocols. However, that is not the focus of this document.

Firewalls play a related role to NAT. A NAT itself provides a form of firewall - it prevents inbound communications unless there was a matching outbound communications first. However, firewalls allow a nearly infinite set of policies about whether a packet will traverse or not. This means that a so-called firewall traversal technique may or may not work depending on the firewall policies. Because of this, the considerations described here are applicable to firewall traversal techniques to the degree that the firewall policy would permit the technique to work. That said, the focus of the document is on NAT, not firewall.

Section 2 provides a simplified view of the network under consideration, sufficient for the purposes of demonstrating the role of each technique. Section 3 is a more detailed description of the various techniques. Section 4 discusses the key architectural

considerations when making the choice. These include security (Section 4.1), deployability (Section 4.2), manageability (Section 4.3) and availability (Section 4.4).

2. Network Model

This section provides a basic model of the network deployments under consideration. It serves several purposes in this specification:

- It serves to delineate the scope of the networks and application protocols under consideration. If an application protocol cannot be mapped into this model, or if a network deployment looks different than this model, the considerations documented here may not apply.

- It serves as a way to help classify the various traversal techniques in terms of their impact on components in the network.

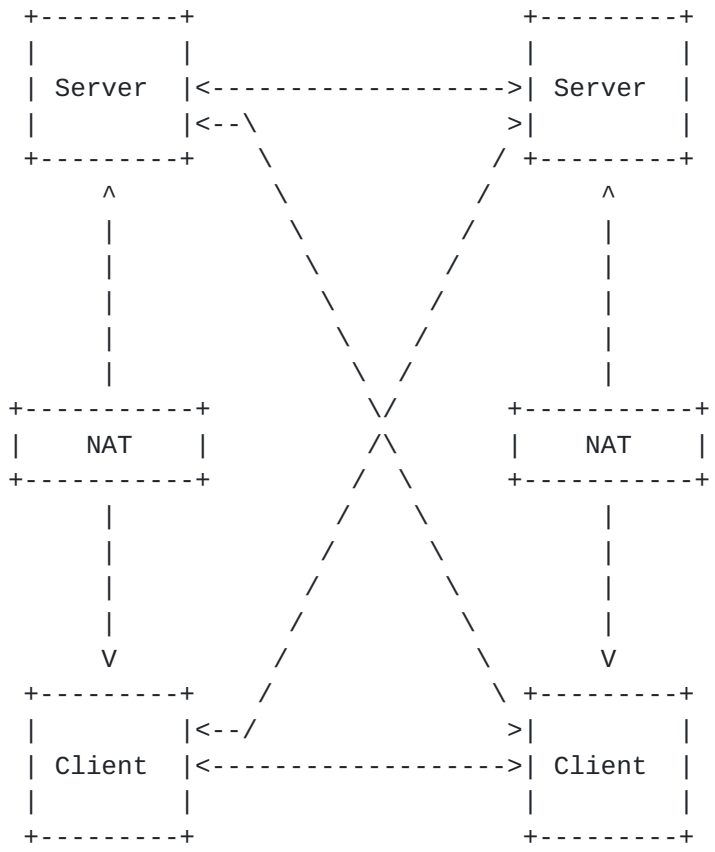

Figure 1

The model under consideration is shown in Figure 1 In this model, the application protocol consists of two classes of entities - clients and servers. Clients refer here to the set of protocol functions that typically reside on end user hosts. In the problem under consideration here, these clients reside behind a NAT or NAT variant, such as Network Address Port Translation (NAPT) (from this point forward, we use the term NAT to include basic NAT and all of its variations). Servers refer here to the set of protocol functions that typically reside in the network. In the problem under consideration here, the servers reside on the public side of the NAT. They are deployed and operated by the service provider. The NAT sits between the clients and servers. It may be owned and deployed by the end user, by the service provider, or by a third entity (typically, the owner of the access network used by the users).

Depending on the application protocol, clients may communicate with each other directly or through servers. Servers may talk with each other or just with clients, also depending on the application protocol.

This model covers application protocols such as FTP, DNS, SIP, RTSP, ITU H.323 and others.

3. Techniques for Traversal

There are many techniques that have been developed to facilitate the traversal of NAT. These techniques can be classified based on the way in which they affect the network in Figure 1. The various techniques ultimately modify the behavior of one or more of the components in the network, and possibly add additional components. With three separate logical components (clients, servers and NAT), there are 7 combinations of modifications that can be made. All seven are actually possible and have been used in actual deployments.

3.1 Modify the NAT: Application Layer Gateways (ALG)

RFC 2663 [1] defines an Application Layer Gateway (ALG) as "application specific translation agents that allow an application on a host in one address realm to connect to its counterpart running on a host in different realm transparently". ALGs are also used to refer to application specific agents in firewalls that allow an application on a host in one network to talk to its counterpart running on a host in another network. One of the key characteristics of the ALG is that it is transparent. It operates without explicit awareness from, or consent by, other entities involved in the application layer protocol.

ALGs appear to have many benefits. By placing them in a small number

of locations in a network, they allow a large number of hosts to use an application which, before deployment of the ALG, did not function. No change is required in the protocols used by those applications. No changes are needed in the clients, and no changes are needed in the servers. The only element that is modified is the element that introduced the problem in the first place - the NAT.

ALGs have seen substantial deployment by enterprises and network operators, particularly for a small number of protocols that were already defined at the time that NAT and firewall usage became commonplace, most notably FTP and DNS.

3.2 Modify the Clients: Unilaternal Self-Address Fixing (UNSAF)

RFC 3424 defines UNSAF as a technique by which a client behind a NAT attempts to discover its address as would be seen in the address realm on the public side of the NAT, and then use that address within application layer protocols instead of its actual IP address. This is done by adding a new element, an "UNSAF server", on the public side of the NAT, and then modifying the client so that it knows how to use the UNSAF server. The actual application protocol server remains unchanged, as does the NAT.

The Simple Traversal of UDP Through NAT [5] is one example of an UNSAF protocol. Teredo [7] is similar to STUN, but focuses on v6 hosts behind a v4 network with NAT that wish to obtain v6 connectivity. Traversal Using Relay NAT (TURN) [6] is a cousin of STUN. However, packets sent to the address provided by the TURN server are actually relayed through the TURN server. TURN is a close cousin of Virtual Private Networks (VPN), of which there are many varieties. TURN differs from VPN in that it operates at a higher layer in the protocol stack, but is similar in that it provides the client with an address and port that route to a server which forwards the packet to the client.

All of the techniques in this section share the property that they require the client software to be modified so that it becomes UNSAF aware. However, this modification tends to be isolated from the application protocol processing itself, with few touch points. Indeed, VPN solutions require no change in the application software itself.

3.3 Modify the Servers: Server Involvement in NAT Navigation (SINN)

The SINN technique involves modifying the servers so that their actual application processing changes, possibly in violation of the specifications they are implementing. However, for some application protocols, the SINN technique allows for NAT traversal without

modifying the clients or the NATs - only the servers. Whether this technique is possible or not depends entirely on the application layer protocol, and usually involves assumptions on client behavior.

One example of the SINN technique are the so-called Session Border Controllers (SBC) in SIP. An SBC operates by ignoring IP addresses and ports provided by the client in its SIP messages, and modifying the IP addresses and ports in SIP messages sent to the clients. These modifications force the media traffic for the session through the SBC, making the SIP network look more like a pure client/server network. SBCs operate only when certain assumptions about client behavior are met - that the client sends and receives its media traffic from the same IP address and port, and that it sends and receives its SIP signaling traffic from the same IP address and port (SIP does not require send and receive address/ports to be the same, though this is common practice).

3.4 Modify the NAT and the Client: RSIP, NSIS

Realm Specific IP (RSIP) [9] can be considered a variant on the UNSAF technique. However, instead of the UNSAF server being outside of the NAT, it is part of the NAT. It is also like VPN in that the client modifications reside below the application processing layer.

Next Steps in Signaling (NSIS) allows clients to directly signal with their NAT to obtain bindings for use in application protocols [10].

3.5 Modify the NAT and the Server: MIDCOM

Middlebox Communications (MIDCOM) [11] allows servers to communicate with the NAT using an application-independent protocol (the MIDCOM protocol [12]) to create and remove NAT bindings. The server can then use the bindings it learned from the NAT to modify the application packets in much the same way an ALG would. In fact, MIDCOM represents a way to build a distributed ALG, by placing the ALG functionality in a protocol server, separate from the NAT itself. In that regards, MIDCOM is a mix between an ALG solution and a SINN solution.

3.6 Modify the Clients and Servers: Protocol Update

In this technique, the protocol gets re-engineered so that it traverses NAT without modification to the NAT itself. This has in fact occurred for several protocols.

3.7 Modify Everything: IPv6

Finally, one can elect to modify the clients and servers, adding

IPv6, and modify the NAT itself by literally removing it from the network based on the presumption that it is no longer needed in IPv6.

4. Considerations for Selection

There are many considerations that must take into account when making a choice. Many of these considerations are not architectural in nature - cost, for example, is a major consideration, but not an architectural one. There are many considerations around selection of a technique within a specific class (i.e., choosing one UNSAF technique over another). These kinds of considerations are important but are difficult to discuss in general, and out of scope for this document. Rather, this section discusses the general architectural considerations around the selection of one of the classes of techniques - ALG, UNSAF, RSIP, NSIS, SINN, MIDCOM, protocol updates and IPv6.

4.1 Security

Security is one of the most important considerations when selecting a technique, and it is one of the biggest differences amongst the techniques. Security issues can be broken broadly into impacts that the technique has on application security, and security concerns from the technique itself.

4.1.1 Undermining Application Security Measures

The reason that the application doesn't work through the NAT is that the application assumes uniquess and reachability properties for its IP addresses. However, the intermediary has broken those properties. Furthermore, reachability through the intermediary requires creation of a translation or binding in that intermediary, which doesn't normally exist based on the application-unaware processing logic in the intermediary. This condition arises when an application protocol includes IP addresses or hostnames within its payloads, and those addresses or hostnames are used by the application protocol as a destination for messages. For such messages to flow, appropriate permissions and/or bindings need to be in place. Thus, all NAT traversal techniques ultimately involve the installation of bindings in the NAT, and the modification of the application protocol messages to reflect the bindings created by the NAT.

In all techniques, the NAT is responsible for installing the needed bindings. However, the various traversal techniques differ in which entities are responsible for modifying the application protocol messages. Performing this modification requires the entity doing so to be able to inspect them and then change them. If the entity that needs to do the inspection and modification is not trusted to do so

by the application protocol, the application protocol itself might provide security techniques which would prohibit the entity from doing so. As such, the technique may require those application security techniques to be disabled, thus undermining the security provided overall.

Unfortunately, these portions of the message - the IP addresses or domain names of hosts to which protocol messages need to be addressed - are exactly the parts of the message that are most critical to protect. Indeed, in the case of DNS, a DNS ALG modifies the primary piece of information that protocols such as DNSSEC were designed to protect [13]. Generally speaking, several classes of attacks can be introduced if these addresses are not protected:

Denial-of-Service: If an attacker can modify these addresses as they transit the network, the attacker can set them to point to a target of a denial-of-service attack. Recipients of these modified messages will direct subsequent messages to this target. Depending on the protocol, this may result in substantial message amplification properties.

Eavesdropping: Instead of modifying the addresses to point to a DoS target, the attacker can modify them to point to themself. Once it receives these messages, the attacker can forward them to the proper recipient. This allows an attacker to eavesdrop the protocol.

Service Disruption: An attacker can modify the addresses so that they route to nowhere (for example, a non-existent IP host). This will likely disrupt operation of the protocol, so that the client receives no service.

Other types of attacks might be possible, depending on the specific protocol.

Worse yet, many application protocols do not provide security that protects only the IP address and port information in a protocol message. Rather, larger parts of the message, if not the entire message, is usually protected. As a result, if the security techniques protecting the IP address and port information are disabled, this will usually disable protection over other, perhaps more critical, parts of the message.

Generally speaking then, the extent to which the traversal technique undermines application layer protocol security measures depends on two factors - whether or not the IP address and port information is protected by security measures in the protocol, and the degree to which the protocol entity is allowed by those measures to modify

those parts of message. In this regard, the entity which should ideally modify the protocol message is the "natural" one - the entity which is responsible for the creation of those parts of the message in the first place. When the "natural" entity modifies the message, its not really a modification at all; the message is created properly in the first place with the IP addresses and ports on the public side of the NAT.

This makes ALGs particularly problematic. Since ALGs reside in the network, as opposed to in the clients or servers, they are never the "natural" entity to modify the protocol message. It is a man-in-the-middle, inspecting and modifying protocol messages without explicit awareness or consent by the actual entities in the protocol. In order for them to function, any application protocol security measures protecting those parts of the message will need to be disabled. The nature of the security functions that need to be disabled differ between firewall and NAT. NAT requires the protocol to be inspected and modified. As such, any security mechanisms providing confidentiality or integrity must be disabled. For firewalls, only inspection is needed. This means that confidentiality mechanisms must be disabled; integrity mechanisms can still function.

Consider the example of SIP. SIP messages carry a MIME payload that describes the multimedia session being established. This payload is usually the Session Description Protocol (SDP) [14]. SDP includes IP address and port information, indicating the transport address where media packets (such as audio and video) should be sent. The SDP is protected by SIP's TLS mechanism, SIP S/MIME, and even digest authentication with the auth-int quality of protection. If these are disabled, SIP's security becomes almost impotent, and many attacks are opened up. For example, if an attacker can access and modify the addresses in the SDP, serious security holes are introduced. The attacker can modify the address to direct the media stream to themself, allowing for eavesdropping of a phone call. Worse yet, an attacker can modify the address across several SIP messages, changing them to point to a target of a distributed denial-of-service attack. This target will receive a steady stream of media packets from the recipients of the modified SIP messages.

As such, ALGs are only appropriate in protocols that lack security in the first place, or in environments where the security measures in a protocol are not needed, such as a closed network environment.

For those protocols where the client is responsible for generating the IP address and port information in the protocol messages (such as for SIP, RTSP, and FTP), and where the application protocol security measures are desired, techniques involving modification of the client

(the UNSAF approaches in Section 3.2) are least intrusive, and do not undermine the application protocol security.

4.1.2 Vulnerabilities Inherent in Traversal Technique

Section 4.1.1 considers security vulnerabilities introduced by NAT traversal techniques because of the need to disable security features in application protocols. Another class of attacks are possible depending on the way in which the traversal technique works.

When the traversal technique is accomplished using a protocol, such as STUN, TURN, ICE, RSIP and MIDCOM, the specifications for those techniques include their security considerations. Many of these security considerations are around secure creation of bindings in a NAT. The bindings are what permit packets from the outside to be routed to a particular host on the inside. They represent the principal resource that needs to be protected by traversal security techniques. Attacks become possible when bindings can be created by unauthorized entities. Denial of service attacks can be launched if the port space in a NAT is exhausted with unauthorized bindings, for example. Numerous attacks are possible when a client believes it has obtained a binding that maps to itself, when it fact it maps to another client. These attacks are discussed in Section 12.1 of RFC 3489 [5].

Generally speaking, the attacks documented in Section 12.1 of RFC 3489 are applicable to any traversal technique where the IP address and port provided to the client are obtained from the source address and port seen by the server, and where the address cannot be security verified before use. This applies to STUN and Teredo, though the usage of these techniques with ICE obviates this risk, since they are validated before use. UNSAF techniques using relays, such as VPN and TURN, do not suffer from these attacks.

These attacks may also be possible in SINN techniques depending on the application protocol. In the case of SIP, SINN is susceptible to these same attacks. The server determines the IP address and port to send media traffic to by using the source IP and port of media it receives from the client. A user eavesdropping a call setup request could inject packets with the IP address and port of the target, creating the attacks. These can also be mitigated with ICE, or through the usage of secure media.

ALGs are also susceptible to these attacks, but they are particularly vulnerable since the attacks are easy to launch. An ALG looks for IP addresses present in a protocol message, and allocates a binding or permission that will allow packets from the outside to traverse the intermediary, and be delivered to the address present in the protocol

message. Creation of a binding to an arbitrary address is accomplished by placing that address into the protocol payload. Thus, source address spoofing is not required, as it is for other traversal techniques.

This attack is particularly easy to launch. Consider a client who is browsing the web behind a firewall. The user goes to a website of a malicious provider. That website provides the user with a java application implementing a basic SIP client. This application generates a SIP INVITE message, and sends it back to the web server. The INVITE message has, within its SDP, the IP address and port of a mail server behind the firewall. The ALG within the firewall creates a permission, allowing traffic from the outside, towards the mail server. The provider can then inject a flood of packets towards the mail server, disabling it.

In a similar attack, if an internal server has vulnerable ports - for example, a port used for telnet-based CLI access - the INVITE request can contain the IP address and port of that service on the server. Rather than launching a DoS attack, the malicious provider can then attempt to telnet into the server.

MIDCOM, like ALGs, create the binding based on the IP address and port in the protocol messages, and thus have similar security considerations.

For protocols like RSIP and NSIS, the bindings are created based on protocol exchanges with the NAT. These exchanges generally require return routability to the IP address and port of the client, and thus provide a means to validate the private addresses used in the binding. As such, they do not suffer from these attacks.

4.2 Deployability Considerations

The various techniques differ in their deployability. The primary factors influencing deployability are the ability of the service provider to modify parts of the system, and the number of such elements that need to be modified.

As discussed in Section 2, the assumption here is that the service provider owns the servers. They may or may not own the NATs in the network, and they may or may not have any ability to affect the clients. As such, traversal techniques that only involve modifying the servers (the SINN technique) are the easiest to deploy. Those which require modifying the clients (the UNSAF techniques) may be feasible depending on whether the service provider can specify to its customers a particular version of client software that needs to be used.

However, techniques which require modification of the NAT itself can be very hard to deploy. In many situations, the service provider is not the same as the operator of the access network used by its customers. In such cases, the access provider may be using NAT, and the service provider has no ability to influence the features or configuration of the device. It may be possible that the device supports the needed capabilities (ALG, NSIS, RSIP, or MIDCOM) anyway. This is more likely for those traversal techniques where the logic in the NAT is not application-specific - NSIS, RSIP, MIDCOM and similar protocols. However, in those cases, those capabilities in the NAT need to be activated, and permissions set up for the application component (the client in the case of NSIS and RSIP, or the proxy server in MIDCOM) to control the NAT. The existence of these permissions may hinge on a business relationship between the service provider and the owner of the NAT. When the customers for a service provider come from many different access networks, and thus many different access network providers, this may be difficult to establish.

For ALGs, there is a chicken and egg problem. Typically, vendors of such devices build functions based on customer demand. A large driver for demand is customer deployment and usage of a new application. However, the application cannot be deployed or used without the ALG that the vendor needs to build. Thus, the application doesn't get deployed and the demand for the ALG never manifests. This is a non-issue for protocols with ALGs widely supported in NAT, including FTP and DNS.

Control is not the only factor impacting deployability; the number of such components is also a factor. For example, if a service provider is offering services to large enterprises, the enterprise is the owner of the NAT. However, the service provider may have a relatively small number of enterprise customers, and thus it may not be unreasonable to require them to use a particular version of a NAT product from a particular vendor in order to get services. As such, the barriers to deploying solutions that require changes in NAT is not as high. The problem becomes amplified for a service provider offering services to small enterprises, of which there are a lot more.

4.3 Manageability

When a failure occurs in the usage of an application (for example, a network timeout or other problem), it will often fall on the shoulders of the application provider to determine the cause of the problem. Diagnosing such difficulties in an operational network is a complex problem. As a result, many application layer protocols contain built in features that allow a provider to trace a problem

after it has occurred. As an example, the Real Time Transport Protocol (RTP) [15] provides a feedback mechanism that allows a participant in a media session to track packet loss and latency as received by the recipient. As another example, SIP provides message header fields, such as Via and Warning, that allow for a trace of the flow of a message through a series of elements, along with an indication of detailed error conditions. Such diagnostic indications become increasingly important as the application protocol involves a large number of network elements or domains; the more entities in the picture, the more places something can go wrong, and the more important the need to find out where it went wrong.

Diagnosing network faults is only one aspect of manageability; managing configuations, accounting, performance and security are also key parts of operating a network.

The ability of the service provider to manage the service they are providing is dependent on their ability to collect information from the various elements on the way in which the service is operating, and to correlate that information together for a coherent view on the state of the network. The selection of a NAT traversal technique impacts manageability of the network primarily due to the differing levels of visibility that the service provider will have on the operation of that aspect of the service. Visibility is based on the level of transparency of operation.

Transparency is primarily an issue for ALGs. They represent yet another entity that is application aware, and involved in the processing of an application protocol. It is possible for failures to occur at these elements, just like they can occur within actual elements of the application protocol. When such failures do occur, the transparent nature of the ALGs makes network diagnostics an almost impossible task.

Firstly, it becomes hard to determine where the failure occurred. Because the ALG is transparent, its operation is invisible to protocol entities on either side of it. Indeed, there may even be multiple ALGs inserted between legitimate protocol entities, and this is often undetectable. No logging information or application protocol features would be able to help pinpoint the location (in terms of IP address or network provider) of the element that generated the errored message.

With some application protocols, it might be possible for the ALG to only be "partly transparent", which means that it might attempt to make its presence known to other protocol entities. While this may seem at first glance to be a good idea, it only further complicates the problem. It results in entities that are now participants in the

application protocol for some aspects of the protocol, but not others. This will likely lead to interoperability problems and additional failure cases, because the behavior of the ALG is not actually compliant to the full requirements outlined by the protocol specification.

ALGs also make it hard to determine when a failure occurs. For example, when an application provider deploys a new protocol element, such as a server or intermediary, it knows that such an element has been deployed, and can therefore look for correlations in errors. Thus, if an operator deploys a new server on Wednesday, and it sees a steep rise in support calls that very same day, there is a good chance that the new server is at fault.

However, such correlations become impossible with ALGs. The reason is that, in the problematic cases, the ALGs will be deployed by different organizations than the provider of the application. As an example, if an application provider deploys a Voice over IP application using SIP, its customer might be an enterprise. One of the routers buried deep within the enterprise network might have a SIP ALG which is turned on one day by an enterprise administrator. This represents a change in the application deployment topology, but not one that is known to the application provider. If that ALG introduces errors into the network, the VoIP provider will begin to see problems and receive customer support calls, but have no idea why, all of a sudden, errors started to happen.

Of course, an application provider is dependent on the enterprise for deployment of normal IP infrastructure in order for the VoIP application to work. It is also possible that the enterprise administrator might deploy a traditional router which introduces packet loss, thereby affecting the VoIP application as well. However, this represents a fundamentally different case. Why? Because the router provides a service (routing of IP datagrams) which is well defined and understood by the developers of applications and application protocols. As a result, those protocols are designed to take into account the known failure modes of those elements (packet loss and jitter, for example), and either recover from them, or allow them to be detected so that further diagnosis can occur. An ALG is different. It doesnt provide a well known service, and its failure modes are not well understood, nor can they be taken into account as part of the design of the application protocol.

In essence, the problem is that an ALG runs counter to the very notion of the end-to-end principle. The end-to-end principle pushes intelligence to the edges of the network. However, an ALG places application intelligence back into the network. This alters the very service model provided by the underlying IP network, and as a result,

makes it difficult for applications to be built around a service model which is then ill defined.

4.4 Avaiability

The choice of a traversal technique impacts availability in many ways. Firstly, techniques which add an additional component to the network add an additional point of failure. This is primarily an issue with the UNSAF techniques, all of which use an additional server of some sort to support traversal through the NAT.

Secondly, network availability depends on the proper functioning of the technique itself. For a technique to function, the essential algorithm and protocol needs to be sound, and its implementation needs to be correct. Both of these can be an issue. Why would the algorithm and protocol be unsound? There are two reasons. The first is that the technique may make assumptions that prove to be untrue in the actual deployment. Second, the technique may not be sufficiently well specified and may miss corner cases not conceived by the designers. Let us consider both of these in turn.

Many of the traversal algorithms and techniques are built around assumptions that they make. Frequently, these assumptions are around behaviors in other components in the system. For example, the SINN techniques make assumptions about the behavior of the clients. The UNSAF techniques make assumptions about the operation of the NAT. These assumptions can be wrong, resulting in brittle operation and failure of the network to operate. STUN is particularly sensitive in this regard; its NAT detection algorithm makes many assumptions about the behavior of a NAT and then tries to validate that behavior by black box testing. This has been addressed in a revision of STUN [16], and when used in conjunction with other techniques, such as ICE [8], the overall system makes fewer assumptions about operation of components in the network, and it is therefore less brittle. service providers should carefully consider the assumptions made by the technique, and validate that these assumptions are correct to the greatest extent possible.

Even if the assumptions are sound, the technique can still fail if the technique has not been fully specified. This problem arises for ALGs, since they are not "first class citizens" in the application protocols they are interacting with. As such, their role in the processing of the protocol is not well defined by specifications, leading to the possibility that corner cases are missed, or that the ALG has interactions with the protocol that cannot be easily resolved. Similar issues arise in the SINN-based solutions, as these are based on non-standard behaviors from elements in the network. Operators should carefully consider that the various cases of

interest are fully supported by the traversal technique.

As an example, SIP describes numerous ways in which SDP can be exchanged during a SIP call in the process of negotiation of codec types and parameters. In the vast majority of calls, a basic exchange occurs - one SDP is sent in the SIP INVITE message (this SDP is called the offer), and a SDP in response (called an answer) in the SIP 200 OK message. However, the spec allows numerous other cases that are far less common, and as a result, likely to be overlooked by ALG implementors. Furthermore, extensions to SIP, such as UPDATE [17] add additional cases. Even if both endpoints in a call support this specification, they will not be able to implement it if the ALG does not support it. Of course, the fact that it is not supported by the ALG will be hard to ascertain because of the diagnostic issues described above.

Finally, the implementation of the technique may not be sound. This is an issue not specific to NAT traversal, of course, but there are some considerations, specifically for ALGs. The manufacturers of intermediary devices housing the ALG are typically router and switch vendors. Building such devices requires expertise at the IP layer, and not for any specific application protocol. However, an ALG is fundamentally an application protocol component, and to develop one properly, requires expertise in that application protocol. This would not be a problem if there was only one application protocol. However, the intermediary vendor needs to have expertise for all application protocols for which ALG support is needed. That is a far more daunting task. Indeed, this problem is identified in the Midcom Architecture [11], which attempts to extract the ALG functionality from the intermediary, place it into an application layer entity, and use a protocol (the MIDCOM protocol) to control the intermediary.

5. Conclusions

There are many techniques that have been described for facilitating the traversal of NAT for protocols for which that capability is not inherent. Broadly speaking, those techniques can be characterized by the set of elements which need to be modified from the "normal" behavior in order for the application to work. As such, a service provider wishing to deploy an application has a wide choice about what technique to use. There are many considerations in that selection. Foremost amongst them is security. NAT traversal techniques frequently impair the inherent security features of the application protocol, and techniques where manipulation is done on the element that is trusted to manipulate that aspect of the message are preferred. Deployability is another concern, and different techniques will only be applicable depending on the scope of control and number of elements that need to be affected. The traveral

techniques also impact manageability, in that some techniques may be difficult to diagnose and monitor in certain deployments. Finally, availability is affected by the technique, due to assumptions on the behavior of elements that may be false, incompletely specified behavior, or incomplete implementation.

6. Security Considerations

Security is a critical consideration when selecting a technique for NAT traversal. The various techniques differ substantially in the impact on overall network security. Service providers should consider carefully their security needs when selecting a technique.

7. IANA Considerations

There are no IANA considerations associated with this specification.

8. IAB Members

Internet Architecture Board members at the time of writing of this document are:

Bernard Aboba

Harald Alvestrand

Rob Austein

Leslie Daigle

Patrik Faltstrom

Sally Floyd

Mark Handley

Bob Hinden

Geoff Huston

Jun-ichiro Itojun Hagino

Eric Rescorla

Pete Resnick

9 Informative References

[1] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[2] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[3] Schulzrinne, H., Rao, A. and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

[4] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

[5] Rosenberg, J., Weinberger, J., Huitema, C. and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", RFC 3489, March 2003.

[6] Rosenberg, J., "Traversal Using Relay NAT (TURN)", draft-rosenberg-midcom-turn-06 (work in progress), October 2004.

[7] Huitema, C., "Teredo: Tunneling IPv6 over UDP through NATs", draft-huitema-v6ops-teredo-04 (work in progress), January 2005.

[8] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Multimedia Session Establishment Protocols", draft-ietf-mmusic-ice-03 (work in progress), October 2004.

[9] Borella, M., Grabelsky, D., Lo, J. and K. Taniguchi, "Realm Specific IP: Protocol Specification", RFC 3103, October 2001.

[10] Stiemerling, M., "A NAT/Firewall NSIS Signaling Layer Protocol (NSLP)", draft-ietf-nsis-nslp-natfw-04 (work in progress), October 2004.

[11] Srisuresh, P., Kuthan, J., Rosenberg, J., Molitor, A. and A. Rayhan, "Middlebox communication architecture and framework", RFC 3303, August 2002.

[12] Quittek, J., Stiemerling, M. and P. Srisuresh, "Definitions of Managed Objects for Middlebox Communication",

draft-ietf-midcom-mib-04 (work in progress), January 2005.

[13] Srisuresh, P., Tsirtsis, G., Akkiraju, P. and A. Heffernan, "DNS extensions to Network Address Translators (DNS_ALG)", RFC 2694, September 1999.

[14] Handley, M. and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[15] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[16] Rosenberg, J., "Simple Traversal of UDP Through Network Address Translators (NAT) (STUN)", draft-ietf-behave-rfc3489bis-00 (work in progress), October 2004.

[17] Rosenberg, J., "The Session Initiation Protocol (SIP) UPDATE Method", RFC 3311, October 2002.

Author's Address

Jonathan Rosenberg, Editor
IAB

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.