Network Working Group
Request for Comments: 3027
Category: Informational

M. Holdrege
ipVerse
P. Srisuresh
Jasmine Networks

January 2001

Protocol Complications with the IP Network Address Translator

Status of this Memo

Copyright Notice

Abstract

Many internet applications can be adversely affected when end nodes are not in the same address realm and seek the assistance of an IP Network Address Translator (NAT) enroute to bridge the realms. The NAT device alone cannot provide the necessary application/protocol transparency in all cases and seeks the assistance of Application Level Gateways (ALGs) where possible, to provide transparency. The purpose of this document is to identify the protocols and applications that break with NAT enroute. The document also attempts to identify any known workarounds. It is not possible to capture all applications that break with NAT in a single document. This document attempts to capture as much information as possible, but is by no means a comprehensive coverage. We hope the coverage provides sufficient clues for applications not covered.

Table of Contents

1.0 Introduction

This document requires the reader to be familiar with the terminology and function of NAT devices as described in [NAT-TERM]. In a nutshell, NAT attempts to provide a transparent routing solution to end hosts requiring communication to disparate address realms. NAT modifies end node addresses (within the IP header of a packet) en-route and maintains state for these updates so that datagrams pertaining to a session are transparently routed to the right end-node in either realm. Where possible, application specific ALGs may be used in conjunction with NAT to provide application level transparency. Unlike NAT, the function of ALG is application specific and would likely require examination and recomposition of IP payload.

The following sections attempt to list applications that are known to have been impacted by NAT devices enroute. However, this is by no means a comprehensive list of all known protocols and applications that have complications with NAT - rather just a subset of the list gathered by the authors. It is also important to note that this document is not intended to advocate NAT, but rather to point out the complications with protocols and applications when NAT devices are enroute.

2.0 Common Characteristics of Protocols broken by NAT

[NAT-TERM] and [NAT-TRAD] have sections listing the specific nature of problems and limitations to NAT devices. Some of these limitations are being restated in this section to summarize characteristics of protocols that are broken by NAT.

2.1 Realm-specific IP address information in payload

A wide range of applications fail with NAT enroute when IP packets contain realm-specific IP address or port information in payload. An ALG may be able to provide work around in some cases. But, if the packet payload is IPsec secured (or secure by a transport or application level security mechanisms), the application is bound to fail.

2.2 Bundled session applications

Bundled session applications such as FTP, H.323, SIP and RTSP, which use a control connection to establish a data flow are also usually broken by NAT devices enroute. This is because these applications exchange address and port parameters within control session to establish data sessions and session orientations. NAT cannot know the inter-dependency of the bundled sessions and would treat each session to be unrelated to one another. Applications in this case can fail for a variety of reasons. Two most likely reasons for failures are: (a) addressing information in control payload is realm-specific and is not valid once packet crosses the originating realm, (b) control session permits data session(s) to originate in a direction that NAT might not permit.

When DNS names are used in control payload, NAT device in conjunction with a DNS-ALG might be able to offer the necessary application level transparency, if NAT has no contention with data session orientation. However, using DNS names in place of realm-specific IP addresses may not be an option to many of these applications (e.g., FTP).

When realm-specific addressing is specified in payload, and the payload is not encrypted, an ALG may in some cases be able to provide the work around necessary to make the applications run transparently across realms. The complexity of ALG depends on the application level knowledge required to process payload and maintain state.

2.3 Peer-to-peer applications

Peer-to-peer applications more than client-server based applications are likely to break with NAT enroute. Unlike Client-server applications, Peer-to-peer applications can be originated by any of the peers. When peers are distributed across private and public realms, a session originated from an external realm is just as likely as the session from a host in private realm. External peers will be able to locate their peers in private realm only when they know the externally assigned IP address or the FQDN ahead of time. FQDN name to assigned address mapping can happen only so long as the enroute NAT device supports DNS-ALG. Examples of Peer-to-peer applications include interactive games, Internet telephony and event-based protocols (such as Instant-Messaging).

This is particularly a problem with traditional NAT and may be less of an issue with bi-directional NAT, where sessions are permitted in both directions.

A possible work-around for this type of problem with traditional-NAT is for private hosts to maintain an outbound connection with a server acting as their representative to the globally routed Internet.

2.4 IP fragmentation with NAPT enroute

IP fragmentation with NAPT enroute is not an issue with any single application, but pervades across all TCP/UDP applications. The problem is described in detail in [NAT-TRAD]. Briefly, the problem goes as follows. Say, two private hosts originated fragmented

TCP/UDP packets to the same destination host. And, they happened to use the same fragmentation identifier. When the target host receives the two unrelated datagrams, carrying same fragmentation id, and from the same assigned host address, the target host is unable to determine which of the two sessions the datagrams belong to. Consequently, both sessions will be corrupted.

2.5 Applications requiring retention of address mapping

NAT will most likely break applications that require address mapping to be retained across contiguous sessions. These applications require the private-to-external address mapping to be retained between sessions so the same external address may be reused for subsequent session interactions. NAT cannot know this requirement and may reassign external address to different hosts between sessions.

Trying to keep NAT from discarding an address mapping would require a NAT extension protocol to the application that would allow the application to inform the NAT device to retain the mappings. Alternately, an ALG may be required to interact with NAT to keep the address mapping from being discarded by NAT.

2.6 Applications requiring more public addresses than available

This is a problem when the number of private hosts is larger than the external addresses available to map the private addresses into. Take for example the rlogin service initiated from a host in private realm supported by NAPT. Rlogin service clients use well-known rlogin port 512 as their TCP port ID. No more than one host in private realm can initiate the service. This is a case of trying to use a service that fundamentally requires more public addresses than are available. NAT devices can conserve addresses, but they cannot create more addresses.

3.0 Protocols that cannot work with NAT enroute

3.1 IPsec and IKE

NAT fundamentally operates by modifying end node addresses (within the IP header) en-route. The IPsec AH standard [IPsec-AH] on the other hand is explicitly designed to detect alterations to IP packet header. So when NAT alters the address information enroute in IP header, the destination host receiving the altered packet will invalidate the packet since the contents of the headers have been altered. The IPsec AH secure packet traversing NAT will simply not reach the target application, as a result.

IPsec ESP ([IPsec-ESP]) encrypted packets may be altered by NAT device enroute only in a limited number of cases. In the case of TCP/UDP packets, NAT would need to update the checksum in TCP/UDP headers, when an address in IP header is changed. However, as the TCP/UDP header is encrypted by the ESP, NAT would not be able to make this checksum update. As a result, TCP/UDP packets encrypted in transport mode ESP, traversing a NAT device will fail the TCP/UDP checksum validation on the receiving end and will simply not reach the target application.

Internet Key Exchange Protocol IKE can potentially pass IP addresses as node identifiers during Main, Aggressive and Quick Modes. In order for an IKE negotiation to correctly pass through a NAT, these payloads would need to be modified. However, these payloads are often protected by hash or obscured by encryption. Even in the case where IP addresses are not used in IKE payloads and an IKE negotiation could occur uninterrupted, there is difficulty with retaining the private-to-external address mapping on NAT from the time IKE completed negotiation to the time IPsec uses the key on an application. In the end, the use of end-to-end IPsec is severely hampered anyway, as described earlier.

For all practical purposes, end-to-end IPsec is impossible to accomplish with NAT enroute.

3.2 Kerberos 4

Kerberos 4 tickets are encrypted. Therefore, an ALG cannot be written. When the KDC receives a ticket request, it includes the source IP address in the returned ticket. Not all Kerberos 4 services actually check source IP addresses. AFS is a good example of a Kerberos 4 service which does not. Services which don’t check are not picky about NAT devices enroute. Kerberos tickets are tied to the IP address that requested the ticket and the service with which to use the ticket.

The K4 ticket (response) contains a single IP address describing the interface used by the client to retrieve the ticket from the TGT from the perspective of KDC. This works fine if the KDC is across a NAT gateway and as long as all of the Kerberos services are also across a NAT gateway. The end user on private network will not notice any problems.

There is also the caveat that NAT uses the same address mapping for the private host for the connection between the client and the KDC as for the connection between the client and the application server. A work around this problem would be to keep an arbitrary connection open to remote server during throughout the ticket lifetime, so as not to let NAT drop the address binding. Alternately, an ALG will need to be deployed to ensure that NAT would not change address bindings during the lifetime of a ticket and between the time a ticket is issued to private host and the time the ticket is used by private host.

But, the ticket will be valid from any host within the private realm of NAPT. Without NAPT, an attacker needs to be able to spoof the source IP addresses of a connection that is being made in order to use a stolen ticket on a different host. With NAPT, all the attacker needs to do from the private realm of NAPT is to simply gain possession of a ticket. Of course, this assumes, NAPT private domain is not a trusted network - not surprisingly, since many attacks occur from inside the organization.

3.3 Kerberos 5

Just as with Kerberos 4, Kerberos 5 tickets are encrypted. Therefore, an ALG cannot be written.

In Kerberos 5, the client specifies a list of IP addresses which the ticket should be valid for, or it can ask for a ticket valid for all IP addresses. By asking for an all-IP-addresses ticket or a ticket containing the NAPT device address, you can get krb5 to work with an NAPT device, although it isn’t very transparent (it requires the clients to behave differently than they otherwise would). The MIT krb5 1.0 implementation didn’t have any configurability for what IP addresses the client asked for (it always asked for the set of its interface addresses) and did not interact well with NAT. The MIT krb5 1.1 implementation allows you to put "noaddresses" somewhere in krb5.conf to request all-IP-addresses-valid tickets.

The K5 ticket (response) contains IP addresses, as requested by the client node, from which the ticket is to be considered valid. If the services being accessed with Kerberos authentication are on the public side of the NAT, then the Kerberos authentication will fail because the IP address used by the NAT (basic NAT or NAPT) is not in the list of acceptable addresses.

There are two workarounds in Kerberos 5 both of which reduce the security of the tickets. The first is to have the clients in NAPT private realm specify the public IP address of the NAPT in the ticket’s IP list. But this leads to the same security problem detailed for K4. Plus, it is not obvious for the client in the private domain to find out the public IP Address of the NAPT. That would be a change of application behavior on end-host.

The second method is to remove all IP addresses from the K5 tickets but this now makes theft of the ticket even worse since the tickets can be used from anywhere. Not just from within the private network.

3.4 The X Windowing System and X-term/Telnet

The X Windowing system is TCP based. However, the client-server relationship with these applications is reverse compared to most other applications. The X server or Open-windows server is the display/mouse/keyboard unit (i.e., the one that controls the actual Windows interface). The clients are the application programs driving the Windows interface.

Some machines run multiple X Windows servers on the same machine. The first X Windows server is at TCP port 6000. The first Open Windows server can be at port 6000 or port 2000 (more flexible). We will mainly refer X windowing system for illustration purposes here.

X-term Transmits IP addresses from the client to the server for the purpose of setting the DISPLAY variable. When set the DISPLAY variable is used for subsequent connections from X clients on the host to an X server on the workstation. The DISPLAY variable is sent inline during the TELNET negotiations as

```
DISPLAY=<local-ip-addr>:<server>.<display>
```

where the <local-ip-addr> is retrieved by looking at the local ip address associated with the socket used to connect to <server>. The <server> determines which port (6000 + <server>) should be used to make the connection. <display> is used to indicate which monitor attached to the X server should be used but is not important to this discussion.

The <local-ip-addr> used is not a DNS name because:

- The is no ability for the local machine to know its DNS name without performing a reverse DNS lookup on the local-ip-addr

- There is no guarantee that the name returned by a reverse DNS lookup actually maps back to the local IP address.

- Lastly, without DNSSEC, it may not be safe to use DNS addresses because they can easily be spoofed. NAT and DNS-ALG cannot work unless DNSSEC is disabled.

A common use of this application is people dialing in to corporate offices from their X terminals at home. Say, the X client is running on a host on the public side of the NAT and X server is running on a host on the private side of the NAT. The DISPLAY variable is transmitted inline to the host the X client is running in some way. The process transmitting the contents of the DISPLAY variable does not know the address of the NAT.

If the channel transmitting the DISPLAY variable is not encrypted, NAT device might solicit the help of an ALG to replace the IP address and configure a port in the valid display range (ports 6000 and higher) to act as a gateway. Alternately, NAT may be configured to listen for incoming connections to provide access to the X Server(s), without requiring an ALG. But, this approach increases the security risk by providing access to the X server that would not otherwise be available. As the ALG tampers with the IP addresses it will also not be possible for X Authorization methods other than MIT-MAGIC-COOKIE-1 to be used. MIT-MAGIC-COOKIE-1 is the least secure of all the documented X Authorization methods.

When START_TLS is used there may be client certificate verification problems caused by the NAT depending on the information provided in the certificate.

3.5 RSH/RLOGIN

RSH uses multiple sessions to support separate streams for stdout and stderr. A random port number is transmitted inline from the client to the server for use as stderr port. The stderr socket is a connection back from the server to the client. And unlike FTP, there is no equivalent to PASV mode. For traditional NAT, this is a problem as traditional NAT would not permit incoming sessions.

RLOGIN does not use multiple sessions. But the Kerberos protected versions of RSH and RLOGIN will not work in a NAT environment due to the ticket problems and the use of multiple sessions.

4.0 Protocols that can work with the aid of an ALG

This document predominantly addresses problems associated with Traditional NAT, especially NAPT.

4.1 FTP

FTP is a TCP based application, used to reliably transfer files between two hosts. FTP uses bundled session approach to accomplish this.

FTP is initiated by a client accessing a well-known port number 21 on the FTP server. This is called the FTP control session. Often, an additional data session accompanies the control session. By default, the data session would be from TCP port 20 on server to the TCP port client used to initiate control session. However, the data session ports may be altered within the FTP control sessions using ASCII encoded PORT and PASV commands and responses.

Say, an FTP client is in a NAT supported private network. An FTP ALG will be required to monitor the FTP control session (for both PORT and PASV modes) to identify the FTP data session port numbers and modify the private address and port number with the externally valid address and port number. In addition, the sequence and acknowledgement numbers, TCP checksum, IP packet length and checksum have to be updated. Consequently the sequence numbers in all subsequent packets for that stream must be adjusted as well as TCP ACK fields and checksums.

In rare cases, increasing the size of the packet could cause it to exceed the MTU of a given transport link. The packet would then have to be fragmented which could affect performance. Or, if the packet has the DF bit set, it would be ICMP rejected and the originating host would then have to perform Path MTU Discovery. This could have an adverse effect on performance.

Note however, if the control command channel is secured, it will be impossible for an ALG to update the IP addresses in the command exchange.

When AUTH is used with Kerberos 4, Kerberos 5, and TLS, the same problems that occur with X-Term/Telnet occur with FTP.

Lastly, it is of interest to note section 4 of RFC 2428 (FTP extensions for IPv6 and NATs) which describes how a new FTP port command (EPSV ALL) can be used to allow NAT devices to fast-track the FTP protocol, eliminating further processing through ALG, if the remote server accepts "EPSV ALL".

4.2 RSVP

RSVP is positioned in the protocol stack at the transport layer, operating on top of IP (either IPv4 or IPv6). However, unlike other transport protocols, RSVP does not transport application data but instead acts like other Internet control protocols (for example, ICMP, IGMP, routing protocols). RSVP messages are sent hop-by-hop between RSVP-capable routers as raw IP datagrams using protocol number 46. It is intended that raw IP datagrams should be used between the end systems and the first (or last) hop router. However, this may not always be possible as not all systems can do raw network I/O. Because of this, it is possible to encapsulate RSVP messages within UDP datagrams for end-system communication. UDP-encapsulated

RSVP messages are sent to either port 1698 (if sent by an end system) or port 1699 (if sent by an RSVP-enabled router). For more information concerning UDP encapsulation of RSVP messages; consult Appendix C of RFC 2205.

An RSVP session, a data flow with a particular destination and transport-layer protocol, is defined by:

Destination Address - the destination IP address for the data packets. This may be either a unicast or a multicast address.

Protocol ID - the IP protocol ID (for example, UDP or TCP).

Destination Port - a generalized destination port that is used for demultiplexing at a layer above the IP layer.

NAT devices are presented with unique problems when it comes to supporting RSVP. Two issues are:

1. RSVP message objects may contain IP addresses. The result is that an RSVP-ALG must be able to replace the IP addresses based upon the direction and type of the message. For example, if an external sender were to send an RSVP Path message to an internal receiver, the RSVP session will specify the IP address that the external sender believes is the IP address of the internal receiver. However, when the RSVP Path message reaches the NAT device, the RSVP session must be changed to reflect the IP address that is used internally for the receiver. Similar actions must be taken for all message objects that contain IP addresses.

2. RSVP provides a means, the RSVP Integrity object, to guarantee the integrity of RSVP messages. The problem is that because of the first point, a NAT device must be able to change IP addresses within the RSVP messages. However, when this is done, the RSVP Integrity object is no longer valid as the RSVP message has been changed. Therefore an RSVP-ALG will not work when RSVP Integrity Object is used.

4.3 DNS

DNS is a TCP/UDP based protocol. Domain Names are an issue for hosts which use local DNS servers in NAT private realm. DNS name to address mapping for hosts in private domain should be configured on an authoritative name server within private domain. This server would be accessed by external and internal hosts alike for name resolutions. A DNS-ALG would be required to perform address to name conversions on DNS queries and responses. [DNS-ALG] describes DNS-ALG in detail. If DNS packets are encrypted/authenticated per DNSSEC, then DNS_ALG will fail because it won't be able to perform payload modifications.

Applications using DNS resolver to resolve a DNS name into an IP address, assume availability of address assignment for reuse by the application specific session. As a result, DNS-ALG will be required to keep the address assignment (between private and external addresses) valid for a pre-configured period of time, past the DNS query.

Alternately, if there isn't a need for a name server within private domain, private domain hosts could simply point to an external name server for external name lookup. No ALG is required when the name server is located in external domain.

4.4 SMTP

SMTP is a TCP based protocol ([SMTP]), used by Internet email programs such as sendmail to send TCP-based email messages to well-known port 25. The mail server may be located within or outside private domain. But, the server must be assigned a global name and address, accessible by external hosts. When mail server is located within private domain, inbound SMTP sessions must be redirected to the private host from its externally assigned address. No special mapping is required when Mail server is located in external domain.

Generally speaking, mail systems are configured such that all users specify a single centralized address (such as fooboo@company.com), instead of including individual hosts (such as fooboo@hostA.company.com). The central address must have an MX record specified in the DNS name server accessible by external hosts.

In the majority of cases, mail messages do not contain reference to private IP addresses or links to content data via names that are not visible to outside. However, some mail messages do contain IP addresses of the MTAs that relay the message in the "Received: " field. Some mail messages use IP addresses in place of FQDN for debug purposes or due to lack of a DNS record, in "Mail From: " field.

If one or more MTAs were to be located behind NAT in a private domain, and the mail messages are not secured by signature or cryptographic keys, an SMTP-ALG may be used to translate the IP address information registered by the MTAs. If the MTAs have static address mapping, the translation would be valid across realms for long periods of time.

The ability to trace the mail route may be hampered or prevented by NAT alone, without the ALG. This can cause problems when debugging mail problems or tracking down abusive users of mail.

4.5 SIP

SIP (Refer [SIP]) can run on either TCP or UDP, but by default on the same port 5060.

When used with UDP, a response to a SIP request does not go to the source port the request came from. Rather the SIP message contains the port number the response should be sent to. SIP makes use of ICMP port unreachable errors in the response to request transmissions. Request messages are usually sent on the connected socket. If responses are sent to the source port in the request, each thread handling a request would have to listen on the socket it sent the request on. However, by allowing responses to come to a single port, a single thread can be used for listening instead.

A server may prefer to place the source port of each connected socket in the message. Then each thread can listen for responses separately. Since the port number for a response may not go to the source port of the request, SIP will not normally traverse a NAT and would require a SIP-ALG.

SIP messages carry arbitrary content, which is defined by a MIME type. For multimedia sessions, this is usually the Session Description Protocol (SDP RFC 2327). SDP may specify IP addresses or ports to be used for the exchange of multimedia. These may loose significance when traversing a NAT. Thus a SIP-ALG would need the intelligence to decipher and translate realm-relevant information.

SIP carries URL’s in its Contact, To and From fields that specify signaling addresses. These URL’s can contain IP addresses or domain names in the host port portion of the URL. These may not be valid once they traverse a NAT.

As an alternative to an SIP-ALG, SIP supports a proxy server which could co-reside with NAT and function on the globally significant NAT port. Such a proxy would have a locally specific configuration.

4.6 RealAudio

In default mode, RealAudio clients (say, in a private domain) access TCP port 7070 to initiate conversation with a real-audio server (say, located an external domain) and to exchange control messages during playback (ex: pausing or stopping the audio stream). Audio session parameters are embedded in the TCP control session as byte stream.

The actual audio traffic is carried in the opposite direction on incoming UDP based packets (originated from the server) directed to ports in the range of 6970-7170.

As a result, RealAudio is broken by default on a traditional NAT device. A work around for this would be for the ALG to examine the TCP traffic to determine the audio session parameters and selectively enable inbound UDP sessions for the ports agreed upon in the TCP control session. Alternately, the ALG could simply redirect all inbound UDP sessions directed to ports 6970-7170 to the client address in the private domain.

For bi-Directional NAT, you will not need an ALG. Bi-directional NAT could simply treat each of the TCP and UDP sessions as 2 unrelated sessions and perform IP and TCP/UDP header level translations.

The readers may contact RealNetworks for detailed guidelines on how their applications can be made to work, traversing through NAT and firewall devices.

4.7 H.323

H.323 is complex, uses dynamic ports, and includes multiple UDP streams. Here is a summary of the relevant issues:

An H.323 call is made up of many different simultaneous connections. At least two of the connections are TCP. For an audio-only conference, there may be up to 4 different UDP ’connections’ made.

All connections except one are made to ephemeral (dynamic) ports.

Calls can be initiated from the private as well as the external domain. For conferencing to be useful, external users need to be able to establish calls directly with internal users’ desktop systems.

The addresses and port numbers are exchanged within the data stream of the ’next higher’ connection. For example, the port number for the H.245 connection is established within the Q.931 data stream. (This makes it particularly difficult for the ALG, which will be required to modify the addresses inside these data streams.) To make matters worse, it is possible in Q.931, for example, to specify that the H.245 connection should be secure (encrypted). If a session is encrypted, it is impossible for the ALG to decipher the data stream, unless it has access to the shared key.

Most of the control information is encoded in ASN.1 (only the User-User Information within Q.931 Protocol Data Units, or PDUs, is

ASN.1-encoded (other parts of each Q.931 PDU are not encoded). For those unfamiliar with ASN.1, suffice it to say that it is a complex encoding scheme, which does not end up with fixed byte offsets for address information. In fact, the same version of the same application connecting to the same destination may negotiate to include different options, changing the byte offsets.

Below is the protocol exchange for a typical H.323 call between User A and User B. A’s IP address is 88.88.88.88 and B’s IP address is 99.99.99.99. Note that the Q.931 and H.245 messages are encoded in ASN.1 in the payload of an RTP packet. So to accomplish a connection through a NAT device, an H.323-ALG will be required to examine the packet, decode the ASN.1, and translate the various H.323 control IP addresses.

```
   User A                                                   User B
         A establishes connection to B on well-
         known Q.931 port (1720)

         ----------------------------------------------->
         Q.931 Setup caller address = 88.88.88.88
                     caller port    = 1120
                     callee address = 99.99.99.99
                     callee port    = 1720
         <-----------------------------------------------
         Q.931 Alerting
         <-----------------------------------------------
         Q.931 Connect H.245 address = 99.99.99.99
                       H.245 port    = 1092

         User A establishes connection to User B at
         99.99.99.99, port 1092

         <---------------------------------------------->
         Several H.245 messages are exchanged (Terminal
         Capability Set, Master Slave Determination and
         their respective ACKs)

         <-----------------------------------------------
         H.245 Open Logical Channel, channel = 257
                   RTCP address = 99.99.99.99
                   RTCP port    = 1093
         ----------------------------------------------->
         H.245 Open Logical Channel Ack, channel = 257
                   RTP address = 88.88.88.88
                   RTP port    = 2002
                   (This is where User A would like RTP
                    data sent to)
                   RTCP address = 88.88.88.88
                   RTCP port    = 2003
         ----------------------------------------------->
         H.245 Open Logical Channel, channel = 257
                   RTCP address = 88.88.88.88
                   RTCP port    = 2003
         <-----------------------------------------------
         H.245 Open Logical Channel Ack, channel = 257
                   RTP address = 99.99.99.99
                   RTP port    = 1092
                   (This is where User B would like RTP data
                    sent to)
                   RTCP address = 99.99.99.99
                   RTP port     = 1093
```

Also note that if an H.323 Gateway resided inside a NAT boundary, the ALG would have to be cognizant of the various gateway discovery schemes and adapt to those schemes as well. Or if just the H.323 host/terminal was inside the NAT boundary and tried to register with a Gatekeeper, the IP information in the registration messages would have to be translated by NAT.

4.8 SNMP

SNMP is a network management protocol based on UDP. SNMP payload may contain IP addresses or may refer IP addresses through an index into a table. As a result, when devices within a private network are managed by an external node, SNMP packets transiting a NAT device may contain information that is not relevant in external domain. In some cases, as described in [SNMP-ALG], an SNMP ALG may be used to transparently convert realm-specific addresses into globally unique addresses. Such an ALG assumes static address mapping and bi-directional NAT. It can only work for the set of data types (textual conventions) understood by the SNMP-ALG implementation and for a given set of MIB modules. Furthermore, replacing IP addresses in the SNMP payload may lead to communication failures due to changes in message size or changes in the lexicographic ordering.

Making SNMP ALGs completely transparent to all management applications is not an achievable task. The ALGs will run into problems with SNMPv3 security features, when authentication (and optionally privacy) is enabled, unless the ALG has access to security keys. [NAT-ARCH] also hints at potential issues with SNMP management via NAT.

Alternately, SNMP proxies, as defined in [SNMP-APPL], may be used in conjunction with NAT to forward SNMP messages to external SNMP engines (and vice versa). SNMP proxies are tailored to the private domain context and can hence operate independent of the specific managed object types being accessed. The proxy solution will require the external management application to be aware of the proxy forwarder and the individual nodes being managed will need to be configured to direct their SNMP traffic (notifications and requests) to the proxy forwarder.

5.0 Protocols designed explicitly to work with NAT enroute

5.1 Activision Games

Activision Games were designed to be NAT-friendly so as not to require an ALG for the games to work transparently through traditional NAT devices. Game players within a private domain can play with other players in the same domain or external domain. Activision gaming protocol is proprietary and is based on UDP. The address server uses UDP port number 21157 and is expected to be located in the global address realm.

Game players connect to the address server first, and send their private IP address information (such as private IP address and UDP port number) in the initial connect message. The server notes private address information from the connect message and external address information from the IP and UDP headers. The server then sends both the private and external address information of the game player to all the other peer players. At this point, each game player knows the private and public address information of every other peer. Subsequent to this, each client opens up symmetrical direct connection to each other and uses whichever address (private or external) works first.

Now, the clients can have a session directly with other clients (or) they can have session with other clients via the gaming server. The key is to allow reuse of the same (global address, assigned UDP port) tuple used for initial connection to the game server for all subsequent connections to the client. A game player is recognized by one of (private address, UDP port) or (global address, assigned UDP port) by all other peer players. So, the binding between tuples should remain unchanged on NAT, so long as the gaming player is in session with one or multiple other players.

Opening a connection to a game server in external realm from a private host is no problem. All NAT would have to do is provide routing transparency and retain the same private-to-external address binding so long as there is a minimum of one gaming session with an external node. But, an NAPT configuration must allow multiple simultaneous UDP connections on the same assigned global address/port.

The above approach has some problems. For example, a client could try contacting a private address, but that private address could be in use locally, when the private address at some other realm is meant. If the node that was contacted wrongfully has some other service or no service registered for the UDP port, the Activision connect messages are expected to be simply dropped. In the unlikely event, a registered application chooses to interpret the message, the results can be unpredictable.

The readers may refer to Activision for the proprietary, detailed information on the function and design of this protocol.

6.0 Acknowledgements

The authors would like to express sincere thanks to Bernard Aboba, Bill Sommerfield, Dave Cridland, Greg Hudson, Henning Schulzrine, Jeffrey Altman, Keith Moore, Thomas Narten, Vernon Shryver and others that had provided valuable input in preparing this document. Special thanks to Dan Kegel for sharing the Activision games design methodology.

7.0 Security Considerations

The security considerations outlined in [NAT-TERM] are relevant to all NAT devices. This document does not warrant additional security considerations.

8.0 References

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

[NAT-TRAD] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[H.323] ITU-T SG16 H.323, Intel white paper, "H.323 and Firewalls", Dave Chouinard, John Richardson, Milind Khare (with further assistance from Jamie Jason).

[SNMP-ALG] Raz, D., Schoenwaelder, J. and B. Sugla, "An SNMP Application Level Gateway for Payload Address Translation", RFC 2962, October 2000.

[SNMP-APPL] Levi, D., Meyer, P. and B. Stewart, "SNMP Applications", RFC 2573, April 1999.

[NAT-ARCH] Hain, T. "Architectural Implications of NAT", RFC 2993, November 2000.

[SMTP] Postel, J., "Simple Mail Transfer Protocol", STDl 10, RFC 821, August 1982.

[FTP] Postel, J. and J. Reynolds, "File Transfer Protocol (FTP)", STD 9, RFC 959, October 1985.

[SIP] Handley, M., Schulzrinne, H., Schooler, E. and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, March 1999.

[X Windows] Scheifler, B., "FYI on the X Window System", FYI 6, RFC 1198, January 1991.

[RSVP] Braden, R., Zhang. L., Berson. S., Herzog, S. and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[DNS-TERMS] Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

[DNS-IMPL] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987.

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

[IPsec] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[IPsec-ESP] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[IPsec-AH] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[IPsec-DOCS] Thayer, R., Doraswamy, N. and R. Glenn, "IP Security Document Roadmap", RFC 2411, November 1998.

[NAT-SEC] Srisuresh, P., "Security Model with Tunnel-mode IPsec for NAT Domains", RFC 2709, October 1999.

[PRIV-ADDR] Rekhter, Y., Moskowitz, B., Karrenberg, D., G. de Groot, and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[ADDR-BEHA] Carpenter, B., Crowcroft, J. and Y. Rekhter, "IPv4 Address Behaviour Today", RFC 2101, February 1997.

Authors' Addresses

Matt Holdrege
ipVerse
223 Ximeno Ave.
Long Beach, CA 90803

EMail: matt@ipverse.com

Pyda Srisuresh
Jasmine Networks, Inc.
3061 Zanker Road, Suite B
San Jose, CA 95134
U.S.A.

Phone: (408) 895-5032
EMail: srisuresh@yahoo.com

Full Copyright Statement

Acknowledgement

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