Transport Working Group
INTERNET-DRAFT
Obsoletes: RFC 1631
Category: Informational
Expire in six months
February 1998

P. Srisuresh
Lucent Technologies
K. Egevang
Intel Corporation

The IP Network Address Translator (NAT)

<draft-rfced-info-srisuresh-05.txt>

Status of this Memo

Preface

The NAT operation described in this document extends address translation introduced in RFC 1631 and includes a new type of network address and TCP/UDP port translation. In addition, this document corrects the Checksum adjustment algorithm published in RFC 1631 and attempts to discuss NAT operation and limitations in detail.

Abstract

Basic Network Address Translation or Basic NAT is a feature by which IP addresses are mapped from one group to another, transparent to users. Network Address Port Translation, or NAPT is an extension to Basic NAT, in that many network addresses and their TCP/UDP ports are translated to a single network address and its TCP/UDP ports.

Together, these two operations, traditionally referred to as NAT, provide a mechanism to connect an isolated routing realm with private unregistered addresses to the external routing network with globally unique registered addresses.

1. Introduction

The need for IP Address translation arises when a network's internal IP addresses cannot be used outside the network either for security reasons or because they are invalid for use outside the network.

Network topology outside a local domain can change in many ways. Customers may change providers, company backbones may be reorganized, or providers may merge or split. Whenever external topology changes with time, address assignment for nodes within the local domain must also change to reflect the external changes. Changes of this type can be hidden from the users within the domain by centralizing changes to a single address translation router.

Basic Address translation feature would allow local hosts on a private network to transparently access the external global network and enable access to selective local hosts from the outside. Organizations with a network setup predominantly for internal use, with a need for occasional external access are good candidates for this feature.

Many Small Office, Home Office (SOHO) users and telecommuting employees have multiple Network nodes in their office, running TCP/UDP applications, but have a single IP address assigned to their remote access router by their service provider to access remote networks. This ever increasing community of remote access users would be benefited by NAPT, which would permit multiple nodes in a local network to simultaneously access remote networks using the single IP address assigned to their router.

There are limitations to using the translation feature. It is mandatory that all requests and responses pertaining to a session be routed via the same NAT router. For this reason, we recommend that NATs be operated on a border router that is unique to a stub domain, where all IP packets are either originated from the domain or destined to the domain. Address translation is predominantly application independent, with the exception of FTP and a few other applications. Encoded FTP sessions and any encoded sessions in general that might include IP addresses in the encoding will not be supported by NAT.

This solution has the disadvantage of taking away the end-to-end

significance of an IP address, and making up for it with increased state in the network. As a result, end-to-end IP network level security assured by IPSec cannot be assumed to end hosts, so long as there exists a NAT router along the route. The advantage of this approach however is that it can be installed without changes to hosts or routers.

2. Terminology and concepts used

2.1. Session flow vs. Packet flow

Connection or session flows are different from packet flows. A session flow indicates the direction in which the session was initiated with reference to a network port. Packet flow is the direction in which the packet has traveled with reference to a network port.

Take for example, an outbound telnet session. The telnet session consists of packet flows in both inbound and outbound directions. Outbound telnet packets carry terminal keystrokes and inbound telnet packets carry screen displays from the telnet server.

Performing address or TCP port translation for a telnet session would involve translation of incoming as well as outgoing packets belonging to that session.

Packets belonging to a TCP/UDP session are uniquely identified by the tuple of (source IP address, source TU port, target IP address, target TU port). Packets belonging to all other sessions are characterized simply by the tuple of (source IP address, target IP address, IP protocol). A session is uniquely identified by the first packet of that session.

2.2. TU ports, Server ports, Client ports

For the reminder of this document, we will refer TCP/UDP ports associated with an IP address simply as "TU ports".

For most TCP/IP hosts, TU port range 0-1023 is used by servers listening for incoming connections. Clients trying to initiate a connection typically select a TU port in the range of 1024-65535. However, this convention is not universal and not always followed. Some client stations initiate connections using a TU port number in the range of 0-1023, and there are servers listening on TU port numbers in the range of 1024-65535.

A complete list of TU port services may be found in Ref[2].

2.3. Start of session for TCP, UDP and others

The first packet of every TCP session tries to establish a session and contains connection startup information. The first packet of a TCP session may be recognized by the presence of SYN bit and absence of ACK bit in the TCP flags. All TCP packets, with the exception of the first packet must have the ACK bit set.

However, there is no deterministic way of recognizing the start of a UDP based session or any non-TCP session.

2.4. Application Level gateway (ALG)

Not all applications lend themselves easily to translation by NATs; especially those that include IP addresses and TCP/UDP ports in the payload. Application Level Gateways (ALGs) are application specific translation agents that allow hosts from one routing realm to connect to hosts in a different realm. The ALGs may optionally utilize address/port assignments by NAT and perform translations of packets pertaining to the application.

3. Overview of NAT

The Address Translation operation presented in this document is called NAT, for Network Address Translator. This is also sometimes referred to as "Traditional NAT", as there are many variations of address translation that lend themselves to different applications. NAT operation described here is a router function that involves
a) dynamic address assignment and address translation or
b) dynamic TCP/UDP port assignment and translation of network address and TCP/UDP port. We will call the former Basic NAT and the latter NAPT. Together they are referred to as NAT. Unless mentioned otherwise, Address Translation or NAT throughout this document will pertain to Basic NAT as well as NAPT. Only the stub border routers as described in figure 1 below may be configured to perform address translation.

```
     \ | /                 .                             /
+---------------+  WAN     .           +-----------------+/
|Regional Router|----------------------|Stub Router w/NAT|---
+---------------+          .           +-----------------+\
                           .                      |         \
                           .                      |  LAN
                           .               ---------------
                     Stub border
```

Figure 1: NAT Configuration

3.1 Overview of Basic NAT

Basic NAT operation is as follows. A stub domain with a set of private network addresses could be enabled to communicate with external network by dynamically mapping to a set of globally valid network addresses. If the number of local nodes are less than or equal to addresses in the global set, each local address is guaranteed to be mapped to an address from global set. Otherwise, local nodes allowed to have simultaneous access to external network are limited by the number of addresses in global set. In addition, individual local addresses may be statically mapped to specific global addresses to ensure guaranteed access to the outside or to expose a local node for access from the outside. Multiple sessions may be initiated from a local node, using the same address mapping.

Addresses inside a stub domain are local to that domain and not valid outside the domain. Thus, addresses inside a stub domain can be reused by any other stub domain. For instance, a single Class A address could be used by many stub domains. At each exit point between a stub domain and backbone, NAT is installed. If there is more than one exit point it is of great importance that each NAT has the same translation table.

```
                                \ | /
                              +---------------+
                              |Regional Router|
                              +---------------+
                            WAN |           | WAN
                                |           |
            Stub A .............|....   ....|............ Stub B
                                |           |
              {s=198.76.29.7,^  |           |  v{s=198.76.29.7,
               d=198.76.28.4}^  |           |  v d=198.76.28.4}
                +-----------------+       +-----------------+
                |Stub Router w/NAT|       |Stub Router w/NAT|
                +-----------------+       +-----------------+
                      |                         |
                      |  LAN               LAN  |
                -------------             -------------
                          |                 |
        {s=10.33.96.5, ^  |                 |  v{s=198.76.29.7,
         d=198.76.28.4}^ +--+             +--+ v d=10.81.13.22}
                         |--|             |--|
                        /____\           /____\
                      10.33.96.5       10.81.13.22
```

Figure 2: Basic NAT Operation

For instance, in the example of figure 2, both stubs A and B internally use class A address 10.0.0.0. Stub A's NAT is assigned the class C address 198.76.29.0, and Stub B's NAT is assigned the class C address 198.76.28.0. The class C addresses are globally unique no other NAT boxes can use them.

When stub A host 10.33.96.5 wishes to send a packet to stub B host 10.81.13.22, it uses the globally unique address 198.76.28.4 as destination, and sends the packet to it's primary router. The stub router has a static route for net 198.76.0.0 so the packet is forwarded to the WAN-link. However, NAT translates the source address 10.33.96.5 of the IP header to the globally unique 198.76.29.7 before the packet is forwarded. Likewise, IP packets on the return path go through similar address translations.

Notice that this requires no changes to hosts or routers. For instance, as far as the stub A host is concerned, 198.76.28.4 is the address used by the host in stub B. The address translations are completely transparent. Of course, this is just a simple example. There are numerous issues to be explored.

3.2. Overview of NAPT

Say, an organization has a private IP network and a WAN link to a service provider. The private network's stub router is assigned a globally valid address on the WAN link and the remaining nodes in the organization have IP addresses that have only local significance. In such a case, nodes on the private network could be allowed simultaneous access to external network, using the single registered IP address with the aid of NAPT. NAPT would allow mapping of tuples of the type (local IP addresses, local TU port number) to tuples of the type (registered IP address, assigned TU port number).

This model fits the requirements of most Small Office Home Office (SOHO) groups to access external network using a single service provider assigned IP address. This model could be extended to allow inbound access by statically mapping a local node per each service TU port of the registered IP address.

In the example of figure 3 below, stub A internally uses class A address 10.0.0.0. The stub router's WAN interface is assigned an IP address 138.76.28.4 by the service provider.

```
                                   \ | /
                                 +-----------------------+
                                 |Service Provider Router|
                                 +-----------------------+
                               WAN |
                                   |
               Stub A .............|....
                                   |
       ^{s=138.76.28.4,sport=1024, |  v{s=138.76.29.7, sport = 23,
       ^ d=138.76.29.7,dport=23}   |  v d=138.76.28.4, dport = 1024}
                       +------------------+
                       |Stub Router w/NAPT|
                       +------------------+
                         |
                         |  LAN
   --------------------------------------------
      |        ^{s=10.0.0.10,sport=3017, |  v{s=138.76.29.7, sport=23,
      |        ^ d=138.76.29.7,dport=23} |  v d=10.0.0.10, dport=3017}
      |                                  |
     +--+      +--+                    +--+
     |--|      |--|                    |--|
    /____\    /____\                  /____\
   10.0.0.1  10.0.0.2   .....        10.0.0.10
```

Figure 3: Network Address Port Translation (NAPT) Operation

When stub A host 10.0.0.10 sends a telnet packet to host 138.76.29.7, it uses the globally unique address 138.76.29.7 as destination, and sends the packet to it's primary router. The stub router has a static route for net 138.76.0.0 so the packet is forwarded to the WAN-link. However, NAPT translates the tuple of source address 10.0.0.10 and source TCP port 3017 in the IP and TCP headers into the globally unique 138.76.28.4 and a uniquely assigned TCP port, say 1024, before the packet is forwarded. Packets on the return path go through similar address and TCP port translations for the target IP address and target TCP port. Once again, notice that this requires no changes to hosts or routers. The translation is completely transparent.

In this setup, only TCP/UDP sessions are allowed and must originate from the local network. However, there are services such as DNS that demand inbound access. There may be other services for which an organization wishes to allow inbound session access. It is possible to statically configure a TU port service on the stub router to be directed to a specific node in the private network.

In addition to TCP/UDP sessions, ICMP messages, with the exception of REDIRECT message type may also be monitored by NAPT router. ICMP query type packets are translated similar to that of TCP/UDP packets, in that the identifier field in ICMP message header will be uniquely mapped to a query identifier of the registered IP address. The identifier field in ICMP query messages is set by Query sender and returned unchanged in response message from the Query responder. So, the tuple of (Local IP address, local ICMP query identifier) is mapped to a tuple of (registered IP address, assigned ICMP query Identifier) by the NAPT router to uniquely identify ICMP queries of all types from any of the local hosts. Modifications to ICMP error messages are discussed in a later section, as that involves modifications to ICMP payload as well as the IP and ICMP headers.

In NAPT setup, where the registered IP address is the same as the IP address of the stub router WAN interface, the router has to be sure to make distinction between TCP, UDP or ICMP query sessions originated from itself versus those originated from the nodes on local network. All inbound sessions (including TCP, UDP and ICMP query sessions) are assumed to be directed to the NAT router as the end node, unless the target service port is statically mapped to a different node in the local network.

Sessions other than TCP, UDP and ICMP query type are simply not permitted from local nodes, serviced by a NAPT router.

4.0. Translation phases of a session.

There are three phases to Address translation, as follows.

4.1. Address binding:

Address binding is the phase in which a local node IP address is associated with a global address for purposes of translation. For addresses that have static mapping, the binding happens at startup time. Otherwise, a local address is bound to a global address dynamically at the time of session initiation from the local node. Once a local address is bound to a global address, all subsequent sessions originating from the same local address will use the same binding for session based packet translation.

In the case of NAPT, where many local addresses are mapped to a single globally unique address, the binding would be from (local IP addr, TU port#) to a TU port of Registered IP address. As with Basic NAT, this binding is determined at the time of session

initiation.

4.2. Address lookup and translation:

Once address binding is established for a connection setup through a NAT port, all subsequent packets belonging to the same connection will be subject to address lookup (and TU port lookup, in the case of NAPT) for translation purposes.

For outbound packets of a session, the source IP address (and source TU port, in case of NAPT) and related fields (such as IP, TCP, UDP and ICMP header checksums) will undergo translation. For inbound packets of a session, the destination IP address (and destination TU port, in case of NAPT) and related fields such as IP, TCP, UDP and ICMP header checksums) will undergo translation.

4.3. Address unbinding:

Address unbinding is the phase in which a local node IP address is no longer associated with a global address for purposes of translation. When the last session based on an address binding is terminated, it is safe to do the address unbinding after session termination.

The end of a TCP session is detected when FIN is acknowledged by both halves of the session or when either half sets RST bit in TCP flags field. Within a couple seconds after this, the session can be safely assumed to have been terminated. Dynamically bound TCP entries that have not been used for say, 24 hours, should also be safe to delete from the bound list. Dynamically bound non-TCP entries that have not been used for say, 1 minute, should also be safe to delete from the bound list. Session timeouts for TCP and non-TCP sessions could optionally be made user configurable. Another good way to handle session terminations is to timestamp entries and keep them as long as possible and retire the longest idle session when it becomes necessary.

5.0. Packet Translations

NATs are, generally speaking, application independent in that the translations are limited to IP/TCP/UDP/ICMP headers and ICMP error messages only. NATs also do not change the payload of any the packets, as payloads tend to be application specific.

However, there are exceptions to this rule. One of the most popular internet applications FTP would not work by this purist approach of NATs. FTP control session carries in its payload the IP address and TCP port information pertaining to the data session it supports. So, NATs are extended to support FTP application as an exception. Some vendors may choose to expand the function of NAT routers to include other applications requiring modifications in payload.

Keeping NATs application independent implies having to work some of the commonly used utilities (which use IP addresses in payload) around NAT. DNS service is one of them. It is recommended that internal DNS servers maintain mapping of names to IP addresses for internal hosts as well as some external hosts. External DNS servers maintain name mapping for external hosts alone and not for any of the internal hosts. If the local network does not have an internal DNS server, all DNS requests will be directed to external DNS server to find address mapping for the external hosts.

Packets pertaining to NAT managed sessions undergo translation in either direction. Individual packet translation issues are covered in detail in the following sub-sections.

NAT modifications are per packet based and can be very compute intensive, as they involve one or more checksum modifications in addition to simple field translations. Luckily, we have an algorithm below, which makes checksum adjustment to IP, TCP, UDP and ICMP headers very simple and efficient. Since all these headers use a one's complement sum, it is sufficient to calculate the arithmetic difference between the before-translation and after-translation addresses and add this to the checksum. The algorithm below is applicable only for even offsets (i.e., optr below must be at an even offset from start of header) and even lengths (i.e., olen and nlen below must be even). Sample code (in C) for this is as follows.

```
void checksumadjust(unsigned char *chksum, unsigned char *optr,
int olen, unsigned char *nptr, int nlen)
/* assuming: unsigned char is 8 bits, long is 32 bits.
  - chksum points to the chksum in the packet
  - optr points to the old data in the packet
  - nptr points to the new data in the packet
*/
{
  long x, old, new;
  x=chksum[0]*256+chksum[1];
  x=~x & 0xFFFF;
  while (olen)
  {
      old=optr[0]*256+optr[1]; optr+=2;
      x-=old & 0xffff;
      if (x<=0) { x--; x&=0xffff; }
      olen-=2;
  }
  while (nlen)
  {
      new=nptr[0]*256+nptr[1]; nptr+=2;
      x+=new & 0xffff;
      if (x & 0x10000) { x++; x&=0xffff; }
      nlen-=2;
  }
  x=~x & 0xFFFF;
  chksum[0]=x/256; chksum[1]=x & 0xff;
}
```

5.1. Header Manipulations

In Basic NAT model, the IP header of every packet must be modified. This modification includes IP address (source IP address for outbound packets and destination IP address for inbound packets) and the IP checksum.

For TCP/UDP sessions, modifications must include update of checksum in the TCP/UDP headers. This is because TCP/UDP checksum also covers a pseudo header which contains the source and destination IP addresses. As an exception, UDP headers with 0 checksum should not be modified.

In NAPT model, modifications to IP header are similar to that of Basic NAT. For TCP/UDP sessions, modifications must be extended to include translation of TU port (source TU port for outbound packets and destination TU port for inbound packets) in the TCP/UDP header.

Modifications to ICMP and FTP packets are considered separately in the following subsections. ICMP packet modifications section covers modifications to ICMP headers as well.

5.2. FTP sessions

The arguments to the File Transfer Protocol (FTP) PORT command and PASV response include an IP address and a TCP port (in ASCII!). If the IP address in PORT command or PASV response is local to the stub domain, then NAT must substitute this. Because the address and TCP port are encoded in ASCII, this may result in a change in the size of packet. For instance, 10,18,177,42,64,87 is 18 ASCII characters, whereas 193,45,228,137,64,87 is 20 ASCII characters. If the new size is same as the previous, only the TCP checksum needs adjustment as a result of change of data. If the new size is less than or greater than the previous, TCP sequence numbers must also be changed to reflect the change in length of FTP control data portion.

A special table is used to correct the TCP sequence and acknowledge numbers with source port FTP or destination port FTP. The table entries should have source, destination, source port, destination port, delta for sequence numbers and a timestamp. New entries are created only when FTP PORT commands or PASV responses are seen. The sequence number delta may be increased or decreased for every FTP PORT command or PASV response. Sequence numbers are incremented and acknowledge numbers are decremented by this delta.

The sequence number adjustment must be coded carefully, not to harm performance for TCP in general. Of course, if the FTP session is encrypted, PORT command and/or PASV response will fail.

5.3. ICMP packet modifications

All ICMP error messages (with the exception of Redirect message type) will need to be modified, when passed through NAT. The ICMP error message types needing NAT modification would include Destination-Unreachable, Source-Quench, Time-Exceeded and Parameter-Problem. NAT should not attempt to modify a Redirect message type.

Changes to ICMP error message will include a minimum of two address modifications and three checksum modifications. This is because these ICMP messages contain part of the original IP packet in the payload. In order for NAT to be completely transparent to the host, the IP address of the IP header embedded in the payload of the ICMP packet must be modified, the checksum field of the same IP header must correspondingly be modified, and the ICMP header checksum must also

be modified to reflect changes made to the IP header and checksum in the payload. Furthermore, the normal IP header must also be modified.

In a NAPT setup, if the IP message embedded within ICMP happens to be a TCP, UDP or ICMP Query packet, you will also need to modify the appropriate TU port number within the TCP/UDP header or the Query Identifier field in the ICMP Query header.

5.4. IP option handling

An IP datagram with any of the IP options Record Route, Strict Source Route or Loose Source Route would involve IP addresses of the intermediate routers. A NAT intermediate router would simply leave the addresses untranslated and not participate in the processing of these options.

5.5. Applications with IP-address Content

Not All applications lend themselves easily to address translation by NATs. Especially, the applications that carry IP address (and TU port, in case of NAPT) inside the payload. Application Level Gateways, or ALGs must be used to perform translations on packets pertaining to such applications. ALGs may optionally utilize address (and TU port) assignments made by NAT and perform translations specific to the application. Some not so transparent ALGs may choose to perform application specific authentication, logging, filtering and other enhanced functions, not often found with application independent NATs. Often, one or more ALGs are used in a NAT router to complement NAT functionality for a private network.

For example, NAT routers would not translate IP addresses within SNMP payloads. It is not uncommon for an SNMP specific ALG to reside on a NAT router to perform SNMP MIB translations proprietary to the private network.

And, if the payload is encrypted, then it is impossible for NATs or even the ALGs to make the translation.

6. Miscellaneous issues

6.1. Partitioning of Local and Global Addresses

For NAT to operate as described in this draft, it is necessary to partition the IP address space into two parts - the local addresses used internal to stub domain, and the globally unique addresses. Any given address must either be a local

address or a global address. There is no overlap.

The problem with overlap is the following. Say a host in stub A wished to send packets to a host in stub B, but the global addresses of stub B overlapped the local addressees of stub A. In this case, the routers in stub A would not be able to distinguish the global address of stub B from its own local addresses.

6.2. Private address space recommendation

The RFC listed in ref[1] has recommendations on address space allocation for private networks. Internet Assigned Numbers Authority (IANA) has three blocks of IP address space, namely 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16 for private internets. In pre-CIDR notation, the first block is nothing but a single class A network number, while the second block is a set of 16 contiguous class B networks, and the third block is a set of 256 contiguous class C networks.

An organization that decides to use IP addresses in the address space defined above can do so without any coordination with IANA or an Internet registry. The address space can thus be used privately by many independent organizations at the same time, with NAT operation enabled on their border routers.

6.3. Routing Across NAT

The router running NAT should not advertise the local networks to the backbone. Only the networks with global addresses may be known outside the stub. However, global information that NAT receives from the stub border router can be advertised in the stub the usual way.

Typically, the NAT stub router will have a static route configured to forward all external traffic to service provider router over WAN link, and the service provider router will have a static route configured to forward NAT packets (i.e., those whose destination IP address fall within the range of NAT managed global address list) to NAT router over WAN link.

6.4. Private Networks that Span Backbones

In many cases, a private network (such as a corporate network) will be spread over different locations and will use a public backbone for communications between those locations. In this case, it is not desirable to do address translation, both because large numbers of hosts may want to communicate across the backbone, thus requiring large address tables, and because there will be more applications that depend on configured addresses, as opposed to going to a name

server. We call such a private network a backbone-partitioned stub.

Backbone-partitioned stubs should behave as though they were a non-partitioned stub. That is, the routers in all partitions should maintain routes to the local address spaces of all partitions. Of course, the (public) backbones do not maintain routes to any local addresses. Therefore, the border routers must tunnel through the backbones using encapsulation. To do this, each NAT box will set aside one global address for tunneling. When a NAT box x in stub partition X wishes to deliver a packet to stub partition Y, it will encapsulate the packet in an IP header with destination address set to the global address of NAT box y that has been reserved for encapsulation. When NAT box y receives a packet with that destination address, it decapsulates the IP header and routes the packet internally.

6.5. Switch-over from Basic NAT to NAPT

In Basic NAT setup, when local network nodes outnumber global addresses available for mapping (say, a class B local network mapped to a class C global address block), external network access to some of the local nodes is abruptly cut off after the last global address from the address list is used up. This is very inconvenient and constraining. Such an incident can be safely avoided by optionally allowing the Basic NAT router to switch over to NAPT setup for the last global address in the address list. Doing this will guarantee that hosts on local network will have continued, uninterrupted access to the external nodes and services.

7.0. NAT limitations

7.1. Privacy, Security, and Debugging Considerations

Unfortunately, NAT reduces the number of options for providing security. With NAT, nothing that carries an IP address or TU port or information derived from an IP address or TU port (such as the IP/TCP/UDP/ICMP header checksum) can be encrypted. While most application-level encryption should be ok, this prevents encryption of TCP/UDP headers.

NAT takes away the end-to-end significance of IP addresses of the end nodes. As a result, end-to-end IP network level security assured by IPSec will not work for end hosts, so long as there exists a NAT router along the route. IPSec is workable with NAT only so long as IPSec and NAT are implemented on the same router (ex: Gateway to Gateway security or Gateway to end node security based on VPNs).

On the other hand, NAT itself can be seen as providing a kind of privacy mechanism. This comes from the fact that machines on the backbone cannot monitor which hosts are sending and receiving traffic (assuming of course that the application data is encrypted).

The same characteristic that enhances privacy potentially makes debugging problems (including security violations) more difficult. If a host is abusing the Internet in some way (such as trying to attack another machine or even sending large amounts of junk mail or something) it is more difficult to pinpoint the source of the trouble because the IP address of the host is hidden in a NAT router.

7.2. ARP responses to NAT mapped global addresses on a LAN interface

NAT must be enabled only on border routers of a stub domain. The examples provided in the document to illustrate Basic NAT and NAPT have maintained a WAN link for connection to external router (i.e., service provider router) from NAT router. However, if the WAN link were to be replaced by a LAN connection and if part or all of the global address space used for NAT mapping belongs to the same IP subnet as the LAN segment, the NAT router would be expected to provide ARP support for the address range that belongs to the same subnet. Responding to ARP requests for the NAT mapped global addresses with its own MAC address is a must in such a situation with Basic NAT setup. If the NAT router did not respond to these requests, there is no other node in the network that has ownership to these addresses and hence will go unresponded.

This scenario is unlikely with NAPT setup except when the single address used in NAPT mapping is not the interface address of the NAT router (as in the case of a switch-over from Basic NAT to NAPT explained in 6.5 above, for example).

Using an address range from a directly connected subnet for NAT address mapping would obviate static route configuration on the service provider router.

It is the opinion of the authors that a LAN link to a service provider router is not very common. However, vendors may be interested to optionally support proxy ARP just in case.

7.3. Translation of fragmented FTP control packets

Translation of fragmented FTP control packets is tricky when the packets contain "PORT" command or response to "PASV" command. Clearly, this is a pathological case. It may be fine to simply

discard the fragments. Alternately, NAT router could attempt to assemble fragments first and then translate prior to forwarding.

Yet another pathological case would be when each character of packets containing "PORT" command or response to "PASV" is sent in a separate datagram, unfragmented. In this case, NAT would simply have to let the packets through, untranslated.

7.4. Translation of outbound TCP/UDP fragmented packets in NAPT setup

Translation of outbound TCP/UDP fragments (i.e., those originating from private hosts) in NAPT setup are doomed to fail. The reason is as follows. Only the first fragment contains the TCP/UDP header that would be necessary to associate the packet to a session for translation purposes. Subsequent fragments do not contain TCP/UDP port information, but simply carry the same fragmentation identifier specified in the first fragment. 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, it is unable to determine which of the two sessions the datagrams belong to. Consequently, both sessions will be corrupted.

7.5. Negative characteristics:

1. NAT is compute intensive even with the help of a clever checksum adjust algorithm, as each data packet is subject to NAT lookup and modifications. As a result, router forwarding throughput will be slowed considerably.

2. NAT increases the probability of mis-addressing. For example, same local address may be bound to different global address at different times and vice versa. As a result, any traffic flow study based purely on global addresses and TU ports could be confused and might misinterpret the results.

3. NAT breaks certain applications or at least makes them more difficult to run.

   DNS is one of the most commonly used utilities that need to be worked around the limitation of NAT as described in section 5.0. Doing this would ensure that local addresses in private network do not appear in the payload of DNS request and response messages.

   Likewise, SNMP based management applications often require an ALG to translate private addresses to distinguish the various

independent nodes within private network.

4. NAT hides the identity of hosts. This is not to be confused with security however. Security on a router must be relegated to firewall functionality, independent of or in conjunction with NAT operation.

8.0. Current Implementations

Many commercial implementations are available in the industry that adhere to the NAT description provided in this document. Linux public domain software contains NAT under the name of "IP masquerade". FreeBSD public domain software has NAPT implementation running as a daemon. Note however that Linux source is covered under the GNU license and FreeBSD software is covered under the UC Berkeley license.

Both Linux and FreeBSD software are free, so you can buy CD-ROMs for these for little more than the cost of distribution. They are also available on-line from a lot of FTP sites with the latest patches.

9.0. Acknowledgements

The first author Srisuresh would like to express his thanks and sincere gratitude to Der-hwa Gan for the knowledge and insight gained during the many probing discussions they had held. Der-hwa has a wide spread knowledge of routers and applications alike and was instrumental in making the author appreciate the many uses of NATs.

10.0. Security Considerations

Below are some of the security considerations associated with NAT routers.

1. UDP sessions are inherently unsafe. Responses to a datagram could come from an address different from the target address used by sender. Below is a quote from RFC 1123, section 2.3 that confirms this.

   When the local host is multihomed, a UDP-based request/ response application SHOULD send the response with an IP source address that is the same as the specific destination address of the UDP request datagram. The

"specific destination address" is defined in the "IP Addressing" section of the companion RFC [INTRO:1].

NAT implementations that do not track datagrams on a per-session basis but lump states of multiple UDP sessions into a single state could compromise the security even further.

2. Multicast sessions (UDP based) are another source for security weaknesses.

   Say, a host on private network initiated a multicast session. Datagram sent by the the private host could trigger responses in the reverse direction from multiple external hosts. NAT implementations that use a single state to track the multicast responses in a multicast session could potentially be the target of security attacks. This multicast specific security concern, however, is not unique to NAT implementations, and exists across all hosts supporting multicast applications.

3. NAT takes away end-to-end significance of IP addresses, TU ports, etc. and makes up for their loss by maintaining a state for each of the sessions it supports. This type of state management for sessions makes NAT a target for security break-ins that hosts have had to deal with. E.g., SYN attacks.

   In a SYN flood attack, an attacker host sends many SYN packets and does not respond with an ACK to the (SYN | ACK)s sent by the receiving host. As the receiving host is waiting for more and more ACKs, the buffer queue will fill up and the receiving host can no longer accept legitimate connections. This means that attackers can block e-mail, web or any other services that may have been provided by the receiving host.

   When a NAT router is in between the attacker and the target host, NAT is maintaining a state for each new session that attacker is initiating. Each new SYN packet sent by the attacker causes a new buffer to be allocated within NAT for management of that new session. Soon, the buffer queue will fill up and the NAT router can no longer support any legitimate connections. This means that attacker is now able to block all services that may have been provided by any of the private hosts, not just the host that is the target of attack.

   One solution may be for NAT implementations to monitor half-open sessions, and set a ceiling on the maximum number of half-open sessions and free up buffers that were allocated for connections that have been half-open for longer than a

certain time period.

4. End-to-end IP network level security assured by IPSec will not work for end hosts, so long as there exists a NAT router along the route. IPSec is workable with NAT only so long as IPSec and NAT are implemented on the same router (ex: Gateway to Gateway security or Gateway to end node security based on VPNs).

REFERENCES

[1] Rekhter, Y., Moskowitz, B., Karrenberg, D., G. de Groot, and, Lear, E. "Address Allocation for Private Internets", RFC 1918 or its successor.

[2] J. Reynolds and J. Postel, "Assigned Numbers", RFC 1700 or its successor.

[3] R. Braden, "Requirements for Internet Hosts -- Communication Layers", RFC 1122 or its successor.

[4] R. Braden, "Requirements for Internet Hosts -- Application and Support", RFC 1123 or its successor.

[5] F. Baker, "Requirements for IP Version 4 Routers", RFC 1812 or its successor.

[6] J. Postel, J. Reynolds, "FILE TRANSFER PROTOCOL (FTP)", RFC 959 or its successor.

[7] "TRANSMISSION CONTROL PROTOCOL (TCP) SPECIFICATION", RFC 793 or its successor.

[8] J. Postel, "INTERNET CONTROl MESSAGE (ICMP) SPECIFICATION", RFC 793 or its successor.

[9] J. Postel, "User Datagram Protocol (UDP)", RFC 768 or its successor.

[10] J. Mogul, J. Postel, "Internet Standard Subnetting Procedure", RFC 950 or its successor.

[11] Brian carpenter, Jon Crowcroft, Yakov Rekhter, "IPv4 Address Behaviour Today", RFC 2101 or its successor.

Authors' Addresses

Pyda Srisuresh
Lucent technologies
Pleasanton, CA 94588-8519
U.S.A.

Voice: (510) 737-2153
Fax: (510) 737-2110
EMail: suresh@livingston.com

Kjeld Borch Egevang
Intel Denmark ApS

Voice: +45 44530100
Fax: +45 44531415
EMail: kbe@casetech.dk
http: //www.freeyellow.com/members/kbe