NAT and Connections Management Facilities

Maxim Shvecov
St. Petersburg Electrotecnical University, MOEVM
Saint-Petersburg, Russia
gris87@yandex.ru

Abstract

TCP/IP networks use IP version 4 also called IPv4. This protocol associates each network node with 32-bit length IP address. However in 1990s people fell short of such range of IP addresses that is why the technology of network address translation (NAT) has been developed. NAT associates a range of local addresses with one or several global addresses. This approach allowed to plug more and more devices into a single network but it made it impossible to connect nodes if there was a NAT device between them. This article considers possible solutions of this problem and compares them.

INDEX TERMS: IP CONNECTION MANAGEMENT, NAT.

I. INTRODUCTION

NAT technology saves IP addresses by the translation of several inside IP addresses into one address or several global ones. However the technology brings a severe problem with hosts located on different networks divided with a NAT device: such hosts cannot establish a direct connection. That leads to the increase of number of leased addresses for nonessential Internet users' hosts.

An Inside network host accessibility from a global network usually is not a matter of concern. But such functionality could widen network technologies possibilities. Thus, it would make local resources and functions available for external use, for example remote host administration, files exchange or video display without its loading on the local drive. Another useful capability is an access to the workstation at work from the Internet whenever a worker needs it what can highly enlarge their efficiency. Many more ways of use can be imagined but the main idea lies in getting rid of leased addresses for the Internet users. That in its turn will increase the number of available global IP addresses. In future the need in NAT technology will vanish once IPv6 becomes widely used but at the moment we should conform to the present situation.

II. MAIN PART

2. Ways to conquer NAT

As it has been already mentioned once hosts on different local networks get the possibility to establish direct connection, it would essentially reduce the number of globally used addresses and help to solve the main problem with network address translation.

Attempts to solve the problem were made way back and resulted in such variants as NAT Traversal, IP Next Layer and VPN. We developed a new method called Flexible NAT. All the methods are based on the existing network technology and are described below.

2.1 NAT Traversal

NAT Traversal is a mechanism developed by Microsoft for Windows family OS-es which configures NAT gateways. It allows to tell a NAT device what association of network addresses to put into the NAT translation table. However this solution has a number of disadvantages:

- NAT Traversal uses a trusted relation model. So all the applications in the private network have an access to all port associations in NAT table. That can lead to getting packets from an ill-conditioned host.
- Conflict detection during the port association lies with an application. If a port is already used an application should use another one.
- Applications having finished the port association should clean the NAT table. Static port mappings can exist infinitely long time so they ought to be deleted.
- Not all NAT devices support NAT Traversal.

2.2 IP Next Layer

IP Next Layer is an extension of IP protocol. According to this method IP packet header includes not only the leased address of NAT gateway but also the local address of the host behind the gateway. If the two hosts are located in one local network standard IP protocol is used. But in case a packet occurs in output of NAT device then the global address is added to its header. On the one hand, this method helps to increase the IP addresses range essentially, provides the direct connection of hosts and even supports the built-in encryption mechanisms. But on the other hand its deployment leads to a number of significant shortages:

- Large amount of NAT devices just behind the gateway to the Internet considerably enlarges packet headers and therefore reduces effective network load.
- Dedicated routers are needed to add global address to a packet header and to route a packet to its destination node.
- Network applications and DNS should be modified. The target address is a full combination of all addresses so applications should be able to form packets with such headers.

2.3 VPN

VPN provides a virtual local network creation over the global network with the help of virtual tunnel between the two hosts. The data sent over the tunnel is encrypted. The security in VPN technology is provided by authentication and authorization mechanisms and data encryption.

Depending on VPN usage next types of VPN can be figured out:

- Intranet VPN joins several distributed hosts into a single local network.
- Remote Access VPN allows a host to access local network.
- Extranet VPN joins two or more different local networks divided by NAT into a single network.
- Client/Server VPN can combine two hosts into a single local network.

Although VPN can establish connection between hosts on different local networks and provides privacy of transferred data it has some limitation, too:

- VPN deployment suggests the usage of a dedicated NAT router able to establish virtual tunnel connection with another NAT router on another network.
- The complexity of encryption algorithms can significantly reduce data transfer rate.
- Virtual network creation suggests availability of a leased VPN server which is not usually granted by Internet providers due to clients security policy.

3. Flexible NAT approach

Flexible NAT mechanism has just the same possibilities as VPN but without the requirement of specific equipment. Flexible NAT also uses port mapping mechanism on NAT device. Unlike IP Next Layer this protocol does not require network applications' or routing algorithms' modification as it uses the existing architecture. The algorithm itself consists in the following:

- Two or more network hosts should determine ports mapping on NAT device and exchange this information with each other. The best choice for this purpose is a leased server supporting mapping and granting access to this information.
- Then on packet sending to any of the hosts it should be forwarded to the corresponding combination of an address and a port number. That is done by replacement of target port number with the corresponding one on NAT device. Initial target port number should be transmitted along with the packet.
- NAT mechanism will provide packet reaching the target but it will likely be dropped by an operating system due to wrong sequence number.
- Therefore it is necessary to switch the target port to its initial value and to pass the packet to the operating system.

Afore-mentioned algorithm provides transparent NAT devices traversing and host-to-host connections. Flexible NAT varies depending on NAT realization. So there are four NAT realizations:

- Symmetric NAT. Untill recently it was the most spread realization. Its specific peculiarity consists in strict bindings to an address and port number in NAT table so that these address and port number packets only can be received. That is the most paranoid NAT realization providing high security for local network hosts. Still it sometimes complicates system administers' lives a lot. And users' lives as well.
- Full Cone NAT. This NAT realization is a great contrast to the previous one. Incoming packets from any external host will be transmitted and forwarded to the corresponding local host if any its entries in NAT table present. Futhermore, source port number does not matter in this case, it can be 53, 54 or whatever. For instance, if any application run on the computer in the local network initiates UDP packets collecting from host 1.2.3.4 on port number 4444 other hosts like 1.2.3.5 or 1.2.3.6 will be able to send packets here untill the entry is deleted from NAT table. And again, this realization checks only transport protocol, target address and port number, not source address or port number.
- Address Restricted Cone NAT (or just Restricted NAT). It is a medium variant between Symmetric and Full Cone NAT as such routers forward incoming packets from the predefined source address but with any port number.
- Port Restricted Cone NAT (or just Port Restricted NAT). It is just the same variant as Restricted NAT but in this case router forwards packets from the predefined port number but with any source address.

While Full Cone NAT and Port Restricted Cone NAT permit direct packets transmission once port numbers are arranged, Symmetric NAT and Address Restricted Cone NAT allow data collecting from public server only. And public server acts as a re-translator in this case. That means that client and server application parts behavior should be specially defined depending on routers NAT types.

STUN protocol (Simple Traversal of UDP through NAT) was developed to define a router NAT type. Its idea is pretty simple. A client sends an outside server probing UDP requests

containing IP addresses and source and target port numbers. Its is necessary that server uses 2 IP addresses what will be explained further.

Now the process of NAT type detection by STUN follows. Let us say the local client behind NAT has local address 192.168.0.111 and public NAT address 1.2.3.4; STUN server has port number 3478 and two addresses – 11.22.33.1 and 11.22.33.2.

1. The client sends a request on main server address (11.22.33.1) with information about addresses and port numbers: from 192.168.0.111:1055 to 11.22.33.1:3478. Having passed NAT packet is changed as source address is replaced with 1.2.3.4. Its port number is changed or left the same depending on NAT realization. If after the predefined time-out period no answer is received the client concludes its behind a blocking firewall and finishes its work.

2. The server answers the client with a message containing addresses and port numbers too, in our case that is source 11.22.33.1:3478 and target 1.2.3.4:1055. If the client received server answer with the same source address as target address in client request it would decide there were no NAT. Then the server and the client would exchange a pair of request-response packets and check the presence of UDP packet blocking firewall. But as the addresses are different client finds out NAT device presence and goes on with NAT type definition. This server answer also informs client of the server's alternative IP address (11.22.33.2) and port number (3478).

3. After that the client sends second probing request with another source port number and a flag indicating that an answer is expected from the alternative server address 11.22.33.2. If a response comes the client concludes there is Full Cone NAT on the route to the server.

4. If no answer comes the client repeats its previous request but sends it on an alternative STUN server address. If an answer address and port number differ from the ones from the first response it means the request initiated the appearance of new entry in NAT table what is typical for Symmetric NAT only.

5. Finally in case this response address and port number are the same as the first response ones NAT device refer to Restricted Cone type. The last thing to find out is its precise type: Address or Port Restricted. For that the client sends the forth request with a flag indicating the server it should answer with another source port number. If the answer is received it is Address Restricted Cone NAT, in other case NAT refers to Port Restricted Cone.

3.1 Relationship with other protocols

The following diagram illustrates FlexNAT protocol's position into the hierarchy of protocols.

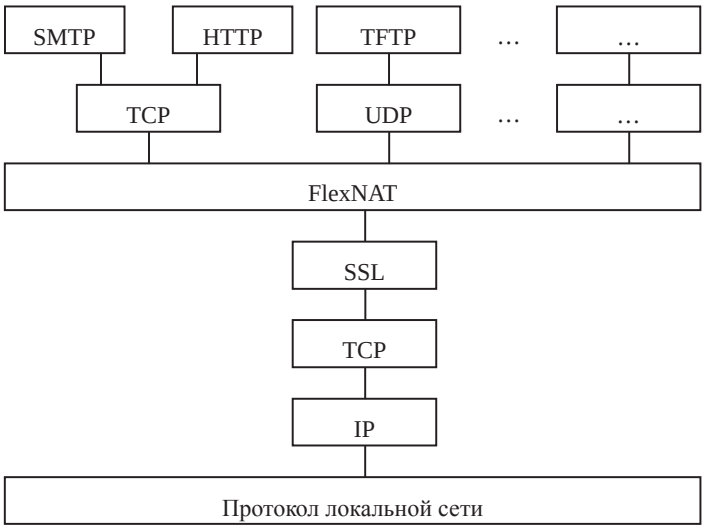


Fig 1. Protocols interaction.

On the one hand FlexNAT protocol interacts with TCP protocol, on the other hand it also interacts with the transport layer protocols. In case of TCP or UDP usage FlexNAT encapsulates original source and target ports, all the other transport layer protocols are fully encapsulated by FlexNAT protocol.

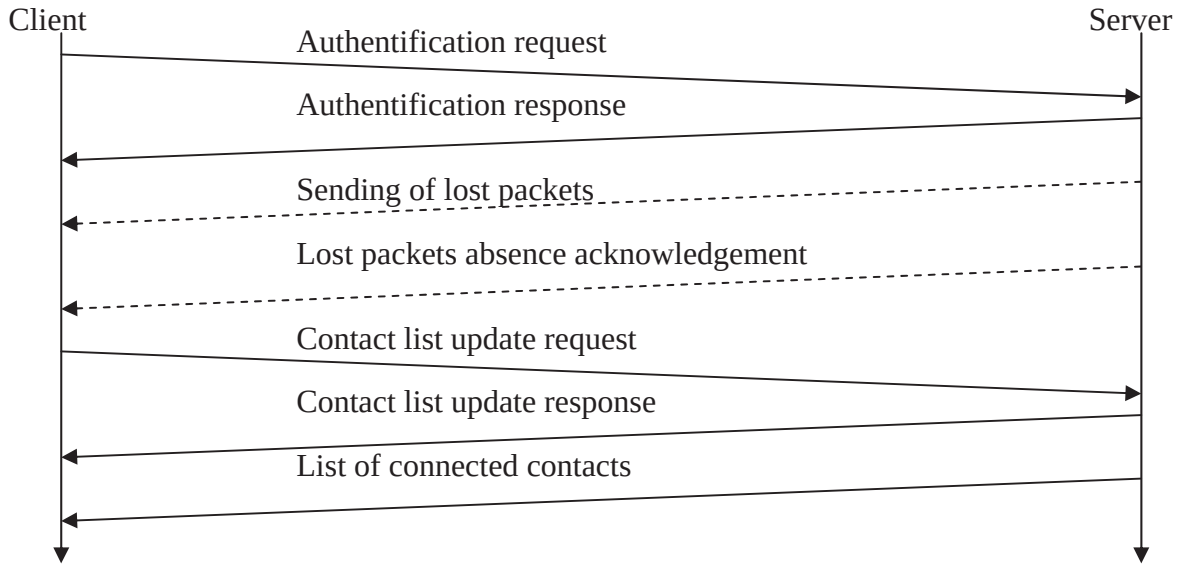
3.2. Protocol description

For the purpose of the correct packet exchange between client nodes, server should be in a global network or be available for nodes, which want to communicate. When client node wants to exchange data with another client node, the former should connect to the server. Thus, a special IP-port record will be added into an address translation table on the NAT-device. Such a record means that particular client node is available through a certain IP-port pair. After that, server forwards to the connected client a list of IP-port records, through which client's contacts are available. Every user's contact is assigned to the virtual IP-address. The process which sends data to the virtual IP, calls a transport protocol program and that protocol in turn calls an internet protocol program. FlexNAT protocol program intercepts such requests, changes destination IP-address, replaces transport protocol with TCP, substitutes destination and source ports with open NAT's ports and adds to the transmitting data original source and destination ports.

3.2.1. User's registration and authorization

If user has no registered account on the server, he is allowed to send a registration request with a desirable password. After receiving such type of message server calculates MD5-hash from the password and that hash is tied to the first free unique identifier (FNID). That user's unique identifier is added to the registration reply packet and sent to the client.

Let's consider a principle of client's authentification on the server by the example of the picture:



After user has typed an identifier and password, authentication request with mentioned identifier and password is sent to a server. Server calculates MD5-hash and compares it with stored one.

If comparison result is negative, server sends client an authentication response with failed-status. Otherwise, if server already has connected user with the same ID, both of them are disconnected and advertised about collision. If comparison result is positive and there is no

already connected user with the same identifier, server checks if lost packets exist for the given user. If no packets exist, server sends authentication response with no-lost-packets-status. Otherwise, client will be advertised about necessity of waiting while all lost packets come. When all lost packets are transferred, server sent client response with all-lost-packets-sent-status.

Client witch received such a packet or packet with no-lost-packets-status, sends contact list update request with a list of MD5-hashes of 32-bits identifiers from the contact list and from the list of supposed contacts. Server compares MD5-hashes and sends client contact list update response.

Then server sends client packet with list of connected contacts, witch consists of all connected at the moment contacts and their internal and external IP-addresses and ports.

3.2.2. Establishing of a trust connection between clients

After user has entered an identifier of a supposed contact, the request packet with receiving contact ID is built on the client side.

When server receives packet, he discovers the receiver and sender and put the receiver ID into the supposed clients list of a sender. After that, the request add-contact-packet with sender ID and some other information is built. If transfer fails, packet is stored on a server. Receiver reads from a packet information about a sender and offers user to add a contact in the contact list. Result of adding a contact is registered in the response packet and sent to a server.

Server determines receiver and sender from a packet and if addition was allowed, puts clients IDs in a contact list and forwards answer to a client requested adding a contact. If transfer fails, packet is stored on a server. When client establishes connection with a server, the last one extracts form received packet source IP and port, through witch one can access node behind the NAT-device. If client's authentication succeeded, server according to the client's contact list and current connected clients transmits a socket list, through witch user's contacts are available, to the client. That information allows user to transfer packets to his contacts through corresponding sockets.

Moreover, after a successful client's authentication every contact from his list is notified about him and his IP and port, through witch he is available.

In case of a connection with a client was terminated, every contact from his list is notified about his disconnection. After receiving such a message nodes should remove information about the socket for disconnected contact.

3.2.3. Virtual IP binding

In order to make an illusion for every client that a packet was sent to the particular contact's port, every user's contact is appointed by his personal virtual IP address.

In case of sending a packet to such an address, destination address is replaced with an address of NAT-device in the IP header and both destination and source ports are replaced with open NAT's ports in the TCP header. Original destination and source ports are stored in the packet's body. The processed packet is sent to the specified client.

If a packet was received from one of the connected at the moment client nodes, then source IP address in the IP header of that packet would be replaced with virtual IP address clinging with the given contact.

Thereby one has an illusion that a packet was sent to the virtual IP address, delivered to a receiver and response comes from the same virtual IP address. Virtual IP addresses have effect only within one node. It is better to use free ranges of IP addresses from a local network, for instance 10.255.255.1-10.255.255.254.

3.2.4. Packet modification before sending to a contact

In order to deliver IP packet to the destination node, it is necessary to do the following:

1. Determine that destination IP address is equal with the virtual IP address of one of connected at the moment contacts. If so, then replace destination IP address with contact's IP address received from a server.
2. If packet isn't a TCP packet, then transform it to TCP. One can reach it by setting the value of 6 (TCP protocol) in the “Protocol” field of an IP header. Moreover it is necessary to add TCP header to the IP datagram's body.
3. Specify the destination port in TCP header with an open port on the NAT-device, behind witch contact is hidden.
4. In order to prevent new address translation correspondence be made by the NAT device when packet goes to the global network, replacement of the source port with an open port on the NAT device is needed.
5. Put down the original value of “Protocol” field in the IP header to the TCP packet's body in order to make possible to restore the last one.
6. If TCP or UDP protocol was used in the original packet, then:
   - Write down into the TCP packet's body original source and destination ports in order to make possible to restore the last one.
   - Write down into the TCP packet's body original TCP or UDP packet body.
7. If a transport protocol different from TCP or UDP was used in the original packet, then it's header and body are written down into the TCP body.

3.2.5. Packet modification after receiving by a contact

After a packet comes to the destination node, it needs to be restored into the original packet and given to an operating system for processing. In order to make the above it is needed to do the following:

1. Find out the contact, from witch packet was received. Such an analysis is based on the source IP address and port. If search succeeds, then replace the source IP address with the virtual IP address of a found contact.
2. Find out from the packet body what kind of transport protocol was used in the original packet and replace the value of a “Protocol” field in the IP header.
3. If TCP protocol was used, then
   a) Replace source and destination ports with those one from the packet body.
   b) Leave the original packet content as is.
4. If UDP protocol was used, then
   a) Restore UDP header, that means to make source and destination packets equal to those one from the packet's body.
   b) Leave the original datagram content as is.
5. In case of using any other protocol, it is necessary to restore it wholly.

3.2.6. Extra Flexible NAT protocol possibilities

SSL protocol provides data protection between transport protocols. At the expense of using SSK protocol, the common key is generated between client and server. That key is used later in the SSL cryptographic algorithms. After a client connects, server gives to the client his SSL certificate. There is no need in the certificate from the client side.

Moreover, SSL protocol provides a huge amount of cryptographic algorithms. After a client gets and approves SSL certificate from a server, common encrypting algorithm is established and encrypting keys exchange is carried out. After above actions made, it is possible to transfer data in the encrypted format.

If user wants to decrease transferred data at expense of hardware utilization, for instance if channel bandwidth is too small, then he can use an integrated data compression mechanism based on Zlib algorithm [RFC 1950]

3.3 Comparison with other protocols: advantages and disadvantages

In contrast to already described methods of direct host-to-host connection establishment FlexNAT protocol:

- does not require specialized equipment;
- does not require such tunings of NAT device as manual mappings establishment and their cleanup when worked finished;
- does not enlarge packet header size on the growth of en-route NAT devices number;
- does not require rewriting of existing protocols;
- does not need a provider.

To summarize FlexNAT features here its advantages and disadvantages come.

Flexible NAT advantages:

- it does not require specialized equipment;
- it provides transferred data privacy;
- it has embedded mechanism of data compression;
- it applies authentication mechanisms;
- leased server enables connection establishment between any hosts.

Flexible NAT disadvantages:

- data transfer rate significantly reduces on data encryption and data compression;
- security of this approach suffers a lot due to public server availability.

So FlexNAT needs extra secure data transfer mechanisms

III. CONCLUSION

IP protocol development basically dictated further network technologies evolution. It is IP protocol that is used by most protocols and applications. But as of the time of its creation the developers could not even imagine this protocol would spread so much. 10 years later it found out that the addressing space was not so big as it seems at first sight. NAT technology was developed to enlarge the addressing space what allowed several local network nodes act as a global network node. Initially NAT technology forbid direct addressing a local network node. Then a number of methods were developed to solve this problem, and they are:

- NAT Traversal,
- IP Next Layer,
- VPN,
- Flexible NAT.

All of them have their own advantages and disadvantages presented in this article. However in future IPv6 protocol is expected to become widely employed as it allows to solve the address lacking problem. In comparison with the previous version IPv6 also has a number of other useful features:

- **IP addresses auto-configuring.**

It simplifies IPv6 networks management greatly.

- **Routing simplification.**

As routers will keep aggregated networks addresses only it will also reduce an average routing table size to 8192 entries.

- **Packet header simplification.**
- **Direct connection between two network nodes.**

As NAT technology blocks direct access to local nodes it does not make sense to use it with IPv6 128-bit addresses.

- **Mobility support.**

As it supplies wireless devices with disconnection-free mechanisms to migrate easily between subnets.

- **QoS support.**
- **Possibility of protocol level cryptographic protection of datagrams.**

REFERENCES

[1] Egevang K., Francis P., "The IP Network Address Translator (NAT)," RFC 1631, Cray Communications, NTT, May 1994.
[2] Deutsch P., Gailly J-L., "ZLIB Compressed Data Format Specification version 3.3," RFC 1950, Aladdin Enterprises, Info-ZIP, May 1996.
[3] Rivest R., "The MD5 Message-Digest Algorithm," RFC 1321, MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992.
[4] Francis P., Gummadi R., “IPNL: A NAT-Extended Internet Architecture”.