NAT Traversal Techniques and Peer-to-Peer Applications

Zhou Hu
Telecommunications Software and Multimedia Laboratory
Helsinki University of Technology
hzhou (at) cc.hut.fi

Abstract

Network Address Translation (NAT) is very useful in Small Office and Home Office (SOHO) community to build a small private network by sharing global routable IP addresses. NAT creates a private IP address realm behind NAT translators. According to common firewall and NAT rule, hosts in private address realm cannot be reached directly from public Internet. In Peer to Peer network, hosts behind NAT gateway have to be reached directly by some way in order to communicate with other peers. NAT techniques hide private hosts thus causing peers not reachable globally. The main reason of the trouble is NAT mangling IP addresses and port numbers thus breaking common end-to-end connections. NAT Traversal Techniques is to let enable end-to-end protocol and application packets through NAT gateway directly or indirectly. NAT is and will be widely adopted over the Internet community, especially in SOHO community. However, NAT technologies are diverse, de facto but not standardized so that the proliferation of NAT devices makes Peer-to-Peer application maker confused and hard to inter-operate with. This paper reviews commonly existing and Peer-to-Peer widely using NAT transversal techniques. I try to give some advise about what should be done on NAT side and Peer-to-Peer application side in order that two things inter-operate smoothly. There is no heal-all technique to make NAT devices fully inter-operable with P2P applications. On the other hand, P2P applications could have to make use general purpose method combined with special efforts to cope with different NAT environment.

KEYWORDS: NAT, P2P, TU hole punching, UPnP, ALG

1 Introduction

Host endpoint can be identified as an IP address or an IP address with a TCP/UDP port number on Internet. Network Address Translation is a technique by which endpoint IP address or IP address with TU port are translated from private address realm to public address realm and back. Network Address Translators (see Fig. 1, both in software or hardware, offer transparent routing to hosts by mapping private and public address realms based on a conceptual communication session.[1, 2, 3] The main reason of NAT's birth is the short-term solution of IPv4 address depletion. In Client-Server network, NATed environment is not a big problem for private addressed clients since they almost always get service by initiating connections with dedicated servers on public addressed Internet.

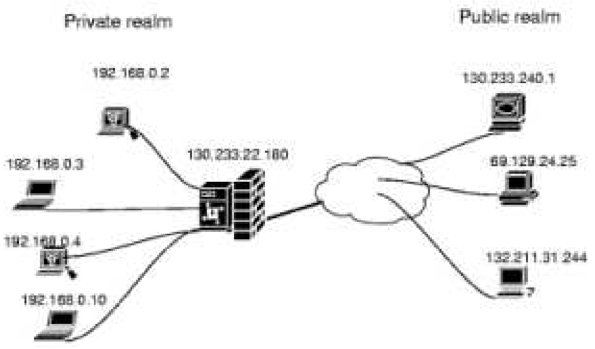

Figure 1: NAT Scenario

Peer-to-Peer network[19, 24] is different with Client-Server network.In Peer-to-Peer network peers have equal positions without classification of client and server and peers are directly connected by other peers. They act both as client and server simultaneously. In NATed environment, general firewall/NAT role does not allow incoming connection to private addressed hosts unless private hosts initiate the connection at first or NAT is specifically configured. The built-in privacy and security benefits of NAT are private addressed hosts hiding. On the other hand, this is a trouble because it is hard to locate and communicate with the private hosts behind a NAT gateway[24, 18]. How two private addressed hosts behind NATs could get to know each other in the very beginning of the Peer-to-Peer communication?

This demands NAT device makers, protocol designers and Peer-to-Peer application vendors to provide smooth and secure two way direct communication including unsolicited incoming connection attempts for customers' hosts residing in NATed environments. Diverse NAT Traversal Techniques offer a variety of NAT devices with transparent traversal abilities to keep the end-to-end connection virtually. Some of them are NAT gateway optimized and plugged techniques such as Universal Plug and Play (UPnP), Application Lever Gateway (ALG). Some of them are fall-back (make use of Client-Server model) approaches, by which they use a relay server or introducer server on either side of NAT gateway or both sides, such as STUN and TCP/UDP hole punching. TU hole punching is the most robust and practical NAT traversal technique[15]. It makes use of a rendezvous server as an introducer for clients behind NAT to get know each other's host endpoints (IP address and TU port).

The NAT's behavior diversity makes protocol designers to avoid leaving IP address and ports in payload of IP packet and to follow some general consideration and guidelines. The diversity also forces Peer-to-Peer application vendors to make application with manual configuration on firewall/NAT or to get clients registered on a relay server firstly. The relationship between NATed environment and Peer-to-Peer application is quite interesting. No single NAT traversal technique is suitable for all NAT necessary environment since NAT is not standarded, that is, no single Peer-to-Peer application could work smoothly in all NATed hosts.

The remainder structure of the paper is: The next section introduces NAT terminology,several NAT traversal techniques, their analysis and usage. The following two sections are advice and guideline for Peer-to-Peer application makers and protocol designers. I close by discussing open problems and security considerations.

2 NAT Traversal Techniques to alleviate the pain

To let Peer to Peer communications between peers smoothly, approaches must allow secure two way communications including unsolicited incoming connection attempts. In his section I will address several approaches to alleviate the pain on P2P networking caused by NAT. They are:

- Universal Plug and Play (UPnP)[11, 12]
- Simple Traversal UDP Through Network Address Translators (STUN)[16]
- Application Level Gateway (ALG)[10]
- UDP/TCP hole punching[13, 15]

These techniques try their best to make NATed environment friendly to applications packet traversal including Peer-to-Peer applications. For some technique, application like clients software could be aware of the NATed environment by automatically querying relay server. For some technique, NAT device are specially designed and configured to allow end-to-end communication.

2.1 NAT terminology

I use the NAT terminology defined in RFC 2663[7] and RFC 3489[16]. A communication session between two hosts is identified as the 4-tuple (local IP, local Port, remote IP, remote Port). The most common NAT is *traditional or outbound* NAT which is an asymmetric mapping between private address realm and public address realm. Outbound NAT by default only permits outbound sessions and blocks all incoming packets unless the session is identified as an existing session initiated from trusted private network host. Pure translation of addresses are *basic* NAT. Both address and ports translation are *Network Address/Port Translation* (NAPT). Obviously outbound NAT conflicts with Peer-to-Peer communication.

Bi-directional NAT or Inbound NAT is to allow sessions initiated from either private or public hosts. A DNS-ALG must be deployed to facilitate NAT for address and domain names mapping.

As you can see, NAT is session based endpoint identities translation. NAT device keeps an endpoint session address binding. Once the first session address binding is created, the subsequent sessions are associated with the same address binding. A new address binding is created by starting a brand new session between local and remote hosts. E.g. 192.168.0.10:1234 requests 130.233.240.9:80 through a NAT 130.233.22.180:6000. If 192.168.0.10:1234 requests 130.233.240.10:80, NAT could allocates 130.233.22.180:6001 by identifying it as a new session. This is called *symmetric* NAT. Symmetric does not reuse session address binding. This results some NAT Traversal techniques failing in traverse packets through NAT devices.

More complex, private hosts might reside in a multi-level NATs or a overlapped NATs when two NAT environments merges with private address realm overlapped.

2.2 Universal Plug and Play, UPnP

UPnP is an architecture and open standard for flexible connectivity of intelligent both wired and wireless devices. The automatic service discovery, addressing, zero configuration are suitable for SOHO. The driving force of UPnP is Microsoft[11]. It is called universal because it is independent on specific operating system, programming languages. It offers universal, flexible connectivity. UPnP's specification is based TCP/IP. UPnP NAT traversal technique is a technique in Windows XP home and professional edition. Windows Internet Connection Sharing (ICS) is UPnP enabled. When a new host needs a connection, UPnP device can automatically configure network addressing, announce its presence on a network subnet, and permit the exchange of device and service descriptions. A Windows XP-based computer can act as an UPnP NAT Gateway, a control point, discovering and controlling devices through a program interface. The application could detect that if it is behind an UPnP-enabled NAT device[12]. Then the application can learn the shared, globally-routable IP address, and configure UDP and TCP port mappings to forward packets from the external ports of the NAT to the internal ports. NAT traversal technique permits peer-to-peer applications to traverse a NAT gateway by dynamically opening and closings ports for communication with other peers. There are two main security vulnerabilities must be patched by Microsoft users. Currently most Internet gateway vendors such as D-Link, Intel, Buffalo Technology, and Arescom are offering UPnP NAT enabled devices[12].

2.3 Simple Traversal UDP Through Network Address Translators, STUN

STUN is lightweight protocol to allow the applications in private address realm to discover the presence of NAT devices and type of NAT between them and public address realm[16]. STUN clients could learn address and port binding used on NATs by sending exploratory STUN request message and receiving STUN response message. Figure 2 shows the deployment of STUN over NAT. STUN is a sim-

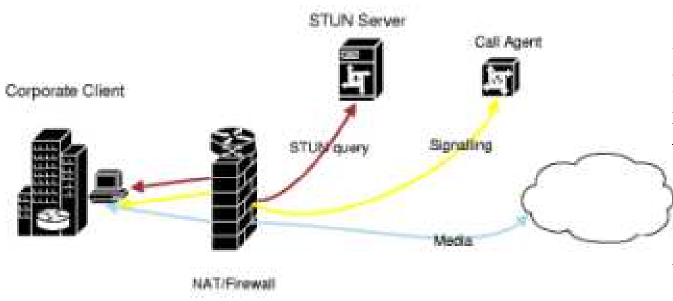

Figure 2: STUN Scenario

ple client-server model protocol. STUN client sends Binding Requests over UDP or Shared Secret Requests on TLS over TCP. In the beginning of the communication session, the shared secret request from the STUN client asks the STUN server for a temporary user name and password for subsequence Binding Requests and Binding Responses' packets integrity check and authentication. The STUN client sends a Binding Request to STUN Server which typically resides in public address realm. The request UDP packet may traverse several NAT devices to arrive STUN server. The Server could learn the last NAT modified source address and port. The STUN server then copies the source address and port into a Binding Response and sends it back to the STUN client. By comparing itself local address and port in response packet, STUN client could learn if it is behind a NAT device. By sending two Binding Requests from the same source address and port to two different IP addresses, STUN client could further learn if it is behind a symmetric NAT device[16]. Unfortunately, STUN does not work with symmetric NAT. Symmetric NAT creates a binding based on source IP address and port as well as destination IP address and port. Client's packet to the application server would result in a new IP address and port binding by symmetric NAT. This fails client's communication with application server or other peers. IETF proposes TURN to solve this problem[21, 17]. The advantage of STUN is it does not require any changes on NAT devices. Clients could learn NAT devices automatically. On the other side, STUN is just a short term solution. It does not work with symmetric NAT which is commonly used by large corporate users. STUN requires client application upgraded to support STUN and an additional STUN server residing in public Internet. Those reasons make STUN deployment slow and unpopular.

2.4 Application Level Gateway, ALG

Network Address Translators only mangle IP addresses and ports in datagram generally. They are not aware of the embedded IP addresses in payload of datagram. Some application embeds IP address and port numbers in packet payload. This kind of packet has problem to traverse NAT devices. E.g. a host IP address instead of a host DNS name in payload. Application Level Gateway always resides in NAT/Firewall devices to modify payload transparently thus work together with NAT to offer transparent routing for packets[10]. ALG could be seen as a NAT extension component. E.g. a DNS ALG could offer translation from host Name-to-Private-Address mapping into Name-to-Public-Address mapping. ALG commonly requires replacement or modification of NAT/Firewall device and configuration. This restricts the deployment of ALG technology[14].

2.5 UDP and TCP hole punching

UDP and TCP hold punching is general purpose, robust technique to establish peer-to-peer communication in Peer-to-Peer network. This technique does not require the application to know the topology of network and presence of NAT devices. It does not modify NAT/Firewall configuration. This technique is introduced in RFC 3027 section 5.1 and specifically analyzed in[25]. The main idea of hole-punching is to have a relaying server which could be reached by clients both or either behind NAT. Figure 3 shows the mechanism of hole punching for two clients behind two different NATs.Two clients send registration message containing private IP address, TU port and public IP address, TU port to relay server. Relay server then introduces two clients to know each other private and public IP address and TU port. UDP hold punching works well for clients behind same NAT, clients behind different NATs and multiple level NATs. NAT has UDP idle timers to keep track of UDP sessions. TCP hole-punching is very similar with UDP hold-punching[15]. TCP hold-punching requires a single local TCP port to listen to an incoming TCP connection and to initiate multiple outgoing TCP connections concurrently. This requires operating system and API support. Unfortunately hole-punching does not work with symmetric NAT also. There is so called port prediction trick, but it still can not guarantee the new allocated port number and introduce new probability of failure. According to Bryan Ford's paper[15], about 82% of the NATs tested support hole punching for UDP, and about 64% support hole punching for TCP streams. Hole-punching has some obvious disadvantages. It turns the part of Peer to Peer network into Client-Server model by introducing a relay server. It adds overhead of bandwidth and increases communication latency.

3 Advice and guideline for Peer-to-Peer Application makers and protocols designers

So far we have a clear picture of different NAT behaviors, the working mechanism of different NAT traversal techniques and their limitation. In this section, I will give some advice and guidelines for protocol designers and Peer-to-Peer application makers.

3.1 Avoid using IP address and TU ports in payload

As RFC 3235[9] implicates, NAT is not aware of embedded IP address and ports so that private unroutable address leaks publicly because NAT would not do proper address mapping. Protocol designers and application makers should avoid using IP address and TU ports in payload directly. In case of

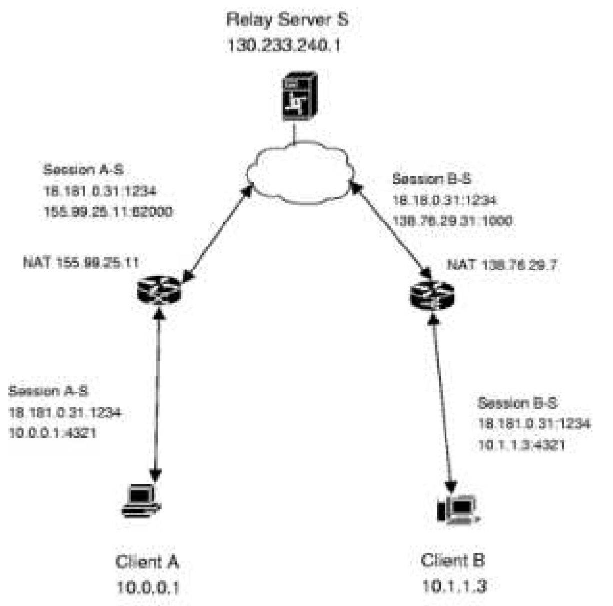

Figure 3: Hole Punching with two clients behind diffferent NAT Scenario

"have to", ALG might help some but also introduces more complexity.

3.2 ALG complexity

Basic NAT, only IP address either static Al or dynamical mapped, would work with Peer-to-Peer application without problems. But the most common deployed NAT in SOHO community is NAPT. With conjection of ALG, NAT could allow inbound connections. This technique would not work without additional software upgrade on NAT device. This increases device cost and decreases the popularity of ALG thus dis-encourage designers to use ALG to make NAT friendly to Peer-to-Peer applications.

3.3 Reuse session address binding

Symmetric NAT does not offer the reuse of session address binding resulting in TU hole punched not possibly being further used for Peer-to-Peer subsequent communication. It is suggested to keep a single connection as possible as it could. NAPT NAT TU timers should be set properly to allow maximum use of session address binding and to recycle TU port resource.

3.4 Peer-to-Peer application automatic and manual configuration

Due the diversity of NAT behaviors, no simple traversal technique could solve all traversal problems. Peer-to-Peer application is suggested to use hybrid approach: combine pure Peer-to-Peer with Client-Server network. For public address realm, hosts works smoothly in Peer-to-Peer network. By use an introducer server to relay the message packet of hosts behinds NAT. Relay server requires intensive computation resource and dynamic directories for dynamic peers. STUN and TU hole punching are this kind of technique. Automatically NAT detection and awareness for peer client is important to diagnose the type of behavior of NAT device, even multi-level NATs. This technique requires at least one dedicated response server to give exploratory response messages to peer clients. Currently, some Peer-to-Peer application has manual configuration ability to let user configure and figure if he/she is behind the NAT by offering several clues of network topology. This is not very suitable for common users without enough network knowledge. As we could notice, general hole punching technique replies on the retention of address mapping or binding. Symmetric NAT does not reuse the session address binding, that is, it does not associate subsequence sessions with the same address binding used in initial session.

4 Security consideration

There is always an engineering trade-off between open and security. Client-Server model is less open than Peer-to-Peer model and it is more secure with NAT and firewall. Currently NAT traversal techniques is to modify NAT device and firewall to let Peer-to-Peer application packets going through. Technique like hole punching opens a two way TU port hole on NAT/Firewall. Naive NAT and Firewall can not simply recognize whether reused session binding is secure or not. NAT breaks the open end-to-end connection. This introduces more complexity of security implementation. One famous example is IPsec can not work over NAT without special configuration.

5 Conclusion and security consideration

Lack of standardization of Network Address Translation technology results in a proliferation of NAT devices. For peer to peer applications, the behavior of NAT devices is highly unpredictable, extremely variable, uncontrollable and hostile[16, 22]. In order to make NAT devices friendly to Peer-to-Peer applications, many NAT traversal techniques are invented. Some technique has limitation resulting in less popularly acceptance and slow deployment. Some technique like hole-punching is widely deployed and general purpose method. For SOHO community, due to the limit knowledge about network and device, management and configuration NAT is pessimistic for end users. UPnP is universally flexible framework natively support by Microsoft Windows and is very suitable for SOHO. ALG is more suitable for large corporate user since it requires software and hardware upgrade and advanced technical maintenance. Most Peer to Peer service chooses hole-punching as a general purpose method to transverse NATs. Relaying server is a fall-back strategy of using Client-Server model to solve peer-to-peer network problem[9, 8]. Using hybrid approach is a make shift and practical way.

From the network point of view, NAT and firewall protect the hosts to initial connection to public Internet by preventing unsolicited incoming connections. NAT and firewall works well in Client-Server model. But they are not suitable for Peer-to-Peer network or violate the open Internet although NAT offer the flexibility of dynamic online hosts[4]. Peer-to-Peer network is to build fully open Internet with equivalent hosts without clear server or client difference. As the Peer-to-Peer applications become more common and popular, we need a common technical solution such as IPv6.

In the short term, NAT will widely exist and there is no single technique which could solve diverse NAT traversal problems. In the long term, IPv6 would offer enough address space to accommodate any IP based terminals. It is said IPv6 could offer each grain of sand a unique address on the planet[5, 6]. At that time, NAT might not be necessary. However those techniques like hole-punching and UPnP is still very useful on IPv6 era.

IETF has formed a new working group, BEHAVE. “This working group proposes to generate requirements documents and best current practices to enable NATs to function in as deterministic a fashion as possible.”[26] There is proposed draft[27] of requirements for NAT behaviors to increase the chance of application working properly.

References

[1] K. Egevang The IP Network Address Translator (NAT) RFC 1631, IETF Network Working Group, May 1994

[2] P. Srisuresh, K. Egevang Traditional IP Network Address Translator (Traditional NAT) RFC 3022, IETF Network Working Group, January 2001

[3] Charles M. Kozierok The TCP/IP Guide. www.tcpioguide.com

[4] Andy Oram Peer-to-Peer: Harnessing the Power of Disruptive Technologies, 1st edition. O’Reilly published, March 2001

[5] R. Hinden, S. Deering IP Version 6 Addressing Architecture RFC 1884, IETF Network Working Group, December 1995

[6] S. Deering, R. Hinden Internet Protocol, Version 6 (IPv6) Specification RFC 2460, IETF Network Working Group, December 1998

[7] P. Srisuresh, M. Holdrege IP Network Address Translator (NAT) Terminology and Considerations RFC 2663, IETF Network Working Group, August 1999

[8] D. Senie NAT Friendly Application Design Guidelines Internet Draft, Work in Progress, July 2000.

[9] D. Senie Netowrk Address Translator (NAT) - Friendly Application Design Guidelines RFC 3235, IETF Network Working Group, January 2002

[10] P. Srisuresh, G. Tsirtsis, P. Akkiraju, A. Heffernan DNS extension to Network Address Translators (DNS_ALG) RFC 2694, IETF Network Working Group, September 1999

[11] Microsoft Corp Understanding UPnP: a White Paper UPnP Forum, June 2000

[12] UPnP Forum UPnP Device Architecture v1.0.1 Draft UPnP Forum, December 2, 2003

[13] B. Ford Network Address Translation and Peer-to-Peer Applications (NATP2P) Internet Draft, Work in Progress, April 2003

[14] T. Hain Architectural Implication of NAT RFC 2993, IETF Network Working Group, November 2000

[15] B. Ford, P. Srisuresh, D. Kegel Peer-to-Peer Communication Across Network Address Translators USENIX Annual Technical Conference, April 2005

[16] J. Rosenberg, J. Weinberger, C. Huitema, R. Mahy STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs) RFC 3489, IETF Network Working Group

[17] Paul Francis Is the Internet Going NUTSS? IEEE Internet Computing, November/December 2003 (Vol. 7, No. 6)

[18] Lisa Phifer The trouble with NAT Core Competence, Cisco System 1992-2001

[19] Clay Shirky What is P2P and What isn’t ? OpenP2P November, 2000

[20] Matei Ripeanu Peer-to-Peer Architecture Case Study: Gnutella Network Technical Report, University of Chicago, 2001.

[21] Saikat Guha, Paul Francis, Takeda Yutaka NUTSS: A SIP-based Approach to UDP and TCP Network Connectivity Sigcomm Future Directions in Network Architecture (FDNA-04) Workshop, August 2004.

[22] Paul Francis, Ramakrishna Gummadi IPNL: A NAT-Extended Internet Architecture SIGCOMM 2001, August, 2001, San Diego

[23] Paul McDougall The power of P2P news, informationweek.com

[24] Dan Kegel NAT and Peer-to-peer networking 1999

[25] M. Holdrege, P. Srisuresh Protocol Complications with the IP Network Address Translator RFC 3027, IETF Network Working Group, January 2001

[26] J Kuthan Behavior Engineering for Hindrance Avoidance (behave) IETF BEHAVE active working group, January 2005

[27] F. Audet, C. Jennings NAT Behavioral Requirements for Unicast UDP Internet-Draft, October 2004