How Practical Are TCP NAT Traversal Schemes?

Chien-Chao Tseng and Chia-Liang Lin

***Abstract*—Peer-to-peer (P2P) communication has emerged as the mainstream of network applications. However, Network Address Translation (NAT) is a barrier to P2P applications and induces NAT traversal problems. TCP NAT traversal is more complicated than that of UDP. Two hosts must perform a three-way handshake procedure to establish a TCP connection and most NAT devices implement TCP state tracking mechanisms to trace TCP stages. This article aims to introduce and verify the applicability of four common TCP NAT traversal schemes including SNT, SLT, P2PNAT and ESi. According to the experiment results, we observe that each TCP NAT traversal scheme generates a specific packet sequence and is suitable for some specific NAT devices. No single scheme works well in every situation since NAT behavior is not standardized. Therefore, there should be a complete behavior test for NAT devices. With NAT behavior information, two hosts can choose a proper scheme for establishing a direct connection.**

***Index Terms*—Network address translation, NAT, NAT Traversal, TCP, TCP NAT traversal.**

I. Introduction

Peer-to-peer (P2P) communication has emerged as the mainstream of network applications and has gained immense popularity in recent years. P2P communication is carried out to avoid the expense and shorten the delay of handling traffic at a server. File sharing is one of the most common P2P applications. However, this style of communication often has problems dealing with Network Address Translation (NAT) [1].

NAT is a solution to IP shortage. It allows more than one computer to share one public IP address. Frankly speaking, NAT is the process of modifying network address information in datagram packet headers while in transit across a traffic routing device. It remaps a given address realm into another.

However, NAT is a barrier to P2P applications and induces NAT traversal problems. Not until an internal host (IH) behind a NAT device sends a packet to the external host (EH) outside the NAT first can the EH send packets to IH directly. This situation becomes worse when both hosts are behind different NAT devices. In other words, NAT devices blocks connection requests originating from the external side and prevents the establishment of P2P connections when both hosts are behind different NAT devices.

NAT traversal [2, 3] is indispensable to removing the barrier of P2P applications. A NAT traversal scheme establishes and maintains TCP/IP network connections across NAT devices. Client-to-client networking applications, especially P2P applications, typically require NAT traversal schemes. With different P2P applications, the underlying transport protocol may be either UDP or TCP. Most VoIP applications adopt UDP as transport protocol while file sharing ones prefer TCP.

TCP NAT traversal is more complicated than that of UDP. Two nodes must perform a three-way handshake procedure to establish a connection. Most NAT devices implement TCP state tracking mechanisms [4] to trace TCP stages. The TCP state tracking mechanism of each NAT device may be different. Accordingly, this article pays more attention to the feasibility of TCP NAT traversal schemes.

Several TCP NAT traversal schemes have been proposed, such as SYN with Normal TTL (SNT) [5], SYN with Low TTL (SLT) [5], Peer-to-Peer NAT (P2PNAT) [2] and Established then SYN-in (ESi) [5]. SNT and SLT evolve from Simple Traversal of UDP Through NATs and TCP too (STUNT) [5]. STUNT is a lightweight protocol and extends Simple Traversal of UDP Through NAT (STUN) [6] to include TCP functionality. It allows IHs to determine the external IP address and port number of a NAT device. It also retrieves packet filtering rules and various timeouts associated with TCP connections through the NAT device. With these parameters, STUNT allows applications to establish TCP connections between two IHs. SLT is a variation of SNT. Instead of using a normal TTL value, a low TTL value of the first TCP SYN packet is set in SLT. P2PNAT takes the advantage of the simultaneous open scenario defined in the TCP specifications. If the TCP SYN packets cross in the network, both IHs respond with TCP SYNACK packets and establish the connection. ESi reuses the existing mappings on the NAT triggered by the IH to establish a connection.

This article aims to introduce and verify the applicability of four common TCP NAT traversal schemes. We conduct a systematic experimental environment in testing each TCP NAT traversal scheme. According to the experiment results, each TCP NAT traversal scheme is suitable for some specific NAT devices. No single scheme works in every situation since NAT behavior is not standardized. Therefore, there should be a complete behavior test for NAT devices. With NAT behavior information, one can choose a proper scheme for establishing a direct connection.

The remainder of this article is organized as follows. We first describe mapping, filtering, TCP filtering and TCP State Tracking rules of a NAT device in detail, and then introduces four common TCP NAT traversal schemes. In the first half of the following section, we describe the setup and design of experiments and network topologies in detail. In the second half, we analyze the experiment results and discuss the application of each traversal scheme. Finally, we summarize our findings and provide suggestions for further research in the final section.

Manuscript received March 5, 2012; revised April 30, 2012.

The authors are with the Department of Computer Science, National Chiao Tung University (e-mail: cllin@cs.nctu.edu.tw; Tel: +886-3-5712121 ext.54792; fax: +886-3-5721490).

II. Literature Review

A. Network Address Translation

NAT allows hosts (IH) in a private network to connect to hosts (EH) in a public network. The usage and toleration of NAT ameliorates IPv4 address depletion by allowing globally registered IP addresses to be either reused or shared among several hosts. Network Address Port Translation (NAPT) is a commonly-adopted NAT implementation, which allows many hosts to share a single IP address through multiplexing streams differentiated by a TCP/UDP port number. In the rest of this paper, NAT refers to NAPT implementation, and a mapped-address is an external global IP address along with a port number allocated by a NAT for a connection attempt from an IH.

The rule of mapped-address allocation is called mapping, while filtering shows how a NAT handles (or discards) packets sent by an EH to an existing mapped-address. The classification of mapping and filtering [7] are described as follows:

B. Mapping

A NAT chooses an external address and maps the port for each connection. The NAT mapping classification then captures these differences in mapping behavior. With the same source address and port at the sender, mapping behavior determines the source public port at the NAT and whether it changes according to the destination address and/or port. Previous authors [7] divided the mapping behavior of NAT into independent mapping, address dependent mapping, and address-and-port dependent mapping. Independent mapping NAT uses the same mapped-address for outbound packets even if there is a change in the destination address or port (0a.i). Address dependent mapping NAT reuses the mapped-address, sending subsequent packets to the same destination address (0a.ii). Finally, address-and-port dependent mapping NAT generates different mapped-address for different destination addresses or ports (0a.iii).

C. Filtering

When the sending packets form the external side of a NAT continue through existing mapped-address to the host (IH) behind the NAT, filtering behavior determines which source addresses (and ports) of hosts (EHs) are able to send packets through the existing mapped-addresses. Previous authors [7] classified NAT filtering behavior into independent filtering, address dependent filtering, and address-and-port dependent filtering. In independent filtering, any EH using any address and any port on the external side can send packets to the IH through the mapped-address (0b.i). Address dependent filtering only accepts packets sent from the same destination address on the external side that created the mapped-address (0b.ii). Finally, address-and-port filtering are limited to the same destination address and port (0b.iii).

D. TCP Filtering

In addition to the above filtering behavior, there are two other kinds of TCP filtering behaviors. TCP filtering behaviors determine how NAT reacts to TCP SYN packets sent by EHs trying to establish connections with IHs. Depending on whether destination address/port is a mapped-address, TCP filtering behaviors can be further classified into ESi and Si filtering behaviors. Both behaviors play important roles in TCP NAT traversal. As 0c.i illustrates, ESi filtering determines how NAT treats incoming TCP SYN packets destined to an existing mapped-address generated by a previous connection. In 0ci, Node A performs a three-way handshake to establish a TCP connection with Node C. Then Node B tries to initiate a three-way handshake with Node A by sending a TCP SYN packet to the mapped-address generated by Node A. If the later incoming TCP SYN packet sent from Node B can traverse NAT X to Node A successfully, NAT X allows “Established then SYN-in” (ESi), otherwise it does not. Si filtering specifies the reaction of a NAT when receiving a TCP SYN packet destined to a non-existing mapped-address. As 0c.ii illustrates, NAT X will filter out unsolicited incoming TCP SYN packet sent from Node B. Furthermore, NATs may respond differently when receiving such unsolicited packets. They may either drop unsolicited incoming TCP SYN packet silently or respond with a TCP RST packet.

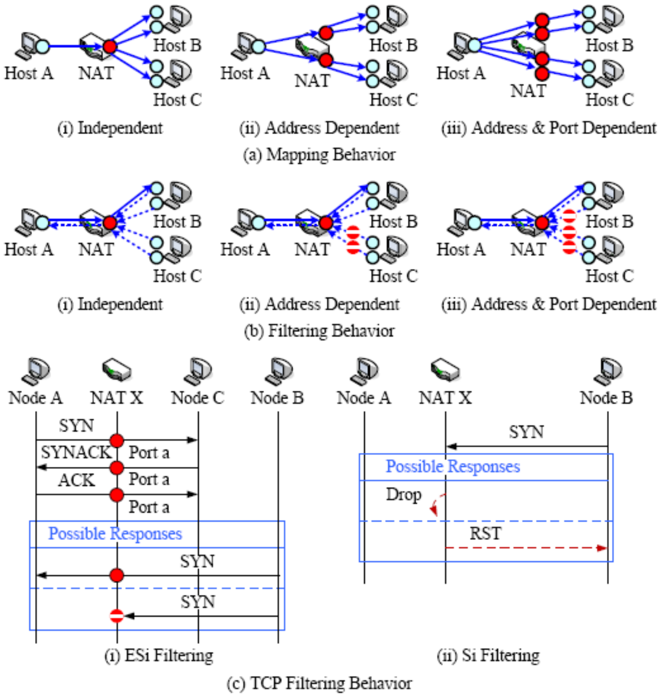


Fig. 1. NAT mapping, filtering and TCP filtering rules: a) mapping; b) filtering; c) TCP filtering.

E. TCP State Tracking

Unlike UDP, TCP sessions are fundamentally unicast in nature and multiple NAT sessions cannot be aggregated. A NAT implements a state machine to track the current phase of each end-to-end TCP connection that traverses the NAT. For example, after receiving an outgoing TCP SYN packet from an IH, the NAT may expect to receive a TCP SYNACK packet from the EH and followed by a TCP ACK packet from the IH. If the NAT does not receive packets with sequence “SYN-out, SYNACK-in, ACK-out”, the mapped-address may be blocked. Furthermore, not every NAT handles all possible packet sequences in the same way. This leads to a specific packet sequence required for TCP NAT traversal may not be acceptable to each NAT. As a result, TCP state tracking makes TCP NAT traversal more complicated and prevents hosts behind different NATs from establishing a direct connection.

III. TCP NAT Traversal

A. SNT (SYN with Normal TTL)

As 0a illustrates, SNT use a TCP SYN packet with normal TTL value. After Node A sends out a normal TCP SYN packet, Node A then aborts the connection attempt and creates a passive TCP socket on the same address and port. Since the first TCP SYN packet is set with a normal TTL value, this packet will traverse NAT X and reach NAT Y. When NAT Y receives an unsolicited TCP SYN packet sent from Node A, NAT Y may drop this packet silently or send back a TCP RST packet. If NAT Y drops the TCP SYN packet silently, NAT X needs to allow packet sequence “SYN-out, SYN-in” for establishing a direct connection. If NAT Y sends back a TCP RST packet, then NAT X has to allow packet sequence “SYN-out, RST-in, SYN-in”. If neither NAT X nor NAT Y allows aforementioned packet sequences, SNT cannot establish a direct connection.

B. SLT (SYN with Low TTL)

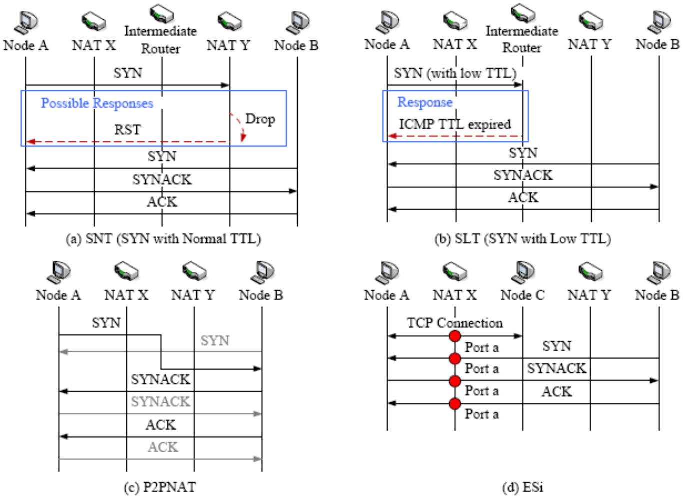
In the SLT approach as 0b illustrates, instead of using a normal TTL value, Node A sends out a TCP SYN packet with low TTL value first. The low-TTL TCP SYN packet is expected to expire somewhere between NAT X and NAT Y. It also created a mapped-address at NAT X. Node A will receive an ICMP TTL–expired packet, aborts the connection attempt and creates a passive TCP socket on the same address and port. Node B then initiates a regular TCP connection to Node A. This approach requires Node A to decide an appropriate TTL value. Besides, NAT X must not consider the ICMP error as a fatal error. That is, NAT X needs to allow packet sequence "SYN-out, TTL-in, SYN-in", which is not a regular sequence of packets. If neither NAT X nor NAT Y allows packet sequence "SYN-out, TTL-in, SYN-in", Node A and Node B cannot establish a direct connection with SLT.

C. P2PNAT

P2PNAT takes the advantage of simultaneous open defined in the TCP specifications. As 0c illustrates, both hosts initiate a connection by sending a TCP SYN packet respectively first. If both TCP SYN packets cross in the network and reach the opposite hosts, both hosts can respond with TCP SYNACK packets to establish the connection successfully. If the TCP SYN packet from Node A arrives at NAT Y before the one from Node B leaves NAT Y, Node A will close its simultaneous open while Node B follows a regular open, vice versa. If both TCP SYN packets cross in the network, NAT X or NAT Y needs to allow packet sequence “SYN-out, SYN-in”. When the TCP SYN packet from Node A arrives at NAT Y before the one from Node B leaves NAT Y, NAT X needs to allow packet sequence “SYN-out, SYN-in” if NAT Y drops unsolicited TCP SYN packet silently. If NAT Y responds with a TCP RST packet, NAT X needs to allow packet sequence “SYN-out, RST-in, SYN-in”.

D. ESi (Established then SYN-in)

As 0d illustrates, if both mapping and filtering behaviors of NAT X are independent, and NAT X allows incoming TCP SYN packet in ESi filtering, then Node A and Node B can use ESi to establish a direct connection, vice versa. In ESi, Node A establishes a TCP connection with Node C and generates a mapped-address on NAT X. Then Node B can establish a direct connection with Node A through the mapped-address on NAT X.



(a) SNT (SYN with Normal TTL)
(b) SLT (SYN with Low TTL)
(c) P2PNAT
(d) ESi

Fig. 2. Common TCP NAT traversal schemes for P2P applications: a) SNT; b) SLT; c) P2PNAT; d) ESi.

IV. Experiment Design and Results

TCP NAT traversal is dispensable to P2P applications adopting TCP as underlying transport protocol. To understand the performance of TCP NAT traversal schemes mentioned above, we design an experimental environment for testing. The performance evaluation metrics include the direct connection ratio (DCR), the connection setup delay, and the resource demand. In the first experiment, we compare the DCR of each traversal scheme among all NAT combinations. In the connection setup delay, we measure the average delay of connection establishment for each TCP NAT traversal scheme. Finally, we calculate the number of messages exchanged during connection setup for estimating resource demand. The following paragraph describes the experiment setup and presents the test results and analysis.

A. Experimental Environment Setup

Conducting the repeated experiments, collecting test results, and verifying the performance of each TCP NAT traversal scheme can be done by hand, but this may waste a lot of manpower and time. Therefore, the experiments in this study use a fully-mesh topology to compare the performance of different TCP NAT traversal schemes. As Fig. 3 and Table 1 illustrate, this topology uses 16 visible domestic NATs available on the market. Node A and Node B are both behind the 16 NATs. Both hosts can switch to the dedicated NATs and thus generates 16*16=256 NAT combinations. However, when Node A and Node B are behind the same NAT, both hosts can connect with each other directly. Therefore, only 256 – 16 = 240 NAT combinations are tested. When testing each NAT combination, both hosts will take turns to initiate a connection. Since TCP is a bidirectional protocol, only one successful initiation will establish a direct connection.

TABLE I: REQUIRED NAT DEVICES IN THE EXPERIMENTS.

Brand	Model	Firmware	Mapping behavior	Filtering Behavior
D-Link	DIR-635	2.32EA	Independent	ESi=D; SI=D
D-Link	DIR-628	1.13EA	Independent	ESi=D; SI=D
D-Link	DIR-615	3.01TW	Independent	ESi=D; SI=D
D-Link	DIR-825	2.00EA	Independent	ESi=D; SI=D
BUFFALO	WZR-AGL300NH	Ver.1.53	Independent	ESi=D; SI=D
PCI	MZK-W300NH	1.02.13	Independent	ESi=D; SI=D
SMC	SMCWBR14S-NL	1.0.2.2, 26-Aug-2009	Independent	ESi=D; SI=D
ZYXEL	NBG-419N/NBG419N	V1.00(BFQ.0)	Independent	ESi=D; SI=D
EDIMAX	BR-6424n	V1.02	Independent	ESi=R; SI=R
Corega	CG-WLBARN80	Ver 1.0.15	Independent	ESi=D; SI=D
PCI	MZK-WNH	1.14	Independent	ESi=R; SI=R
ASUS	RT-N16	1.0.0.6	Independent	ESi=D; SI=D
AboCom	WR5205	v24.5.0.0.6.5	Independent	ESi=R; SI=R
BELKIN	N1 Vision	F5D8232-4_WW_1.00.11	Independent	ESi=D; SI=D
AXIMCom	PGP-116N P2P GEAR PRO	2.0.3 (C.3)	Independent	ESi=R; SI=R
LevelOne	WBR-6001	R1.97g6-R86_20071204_1	Independent	ESi=A; SI=D
(A: Accept; D: Drop; R: RST)				

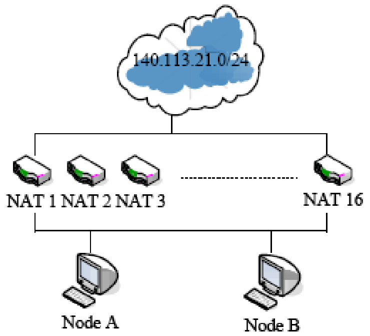


Fig.3. Experiment setup.

TABLE II: DCR, CONNECTIVITY CHECK DELAY AND CONNECTIVITY CHECK MESSAGES OF EACH SCHEME.

	DCR	Connectivity Check Delay (s)	Connectivity Check Messages

STUNT	130/240
	(54.17%)
		Success=0.16;
		Failure=8.15
			4

SLT	114/240
	(47.50%)
		Success=1.16;
		Failure=9.25
			4

ESi	30/240
	(12.50%)
		Success=0.09;
		Failure=8.07
			3

P2PNAT	130/240
	(54.17%)
		Success=0.49;
		Failure=8.93
			6

B. *Experiment Results, Analysis, and Discussion*

SNT

In SNT, the TTL value of first TCP SYN packet is a normal one. Since the first TCP SYN packet is a normal one and the packet will arrive at the opposite NAT, the Si filtering behavior of the opposite NAT and the TCP state tracking of the initiating NAT will affect the direct connection directly. If the initiating NAT allows packet sequence “SYN-out, SYN-in” and the Si filtering behavior of the opposite NAT is silent drop, both hosts can establish a direct connection. If the opposite NAT responds a TCP RST packet, the originating NAT needs to allow packet sequence “SYN-out, RST-in, SYN-in” for establishing a direct connection.

SLT

When performing the experiments, we observe that the originating NAT has to allow packet sequence "SYN-out, TTL-in, SYN-in", or SLT will fail in establishing a direct connection. Moreover, SLT may have problem in finding a proper TTL value for the first TCP SYN packet. When the TTL value sets too low, the TCP SYN packet cannot pass the initiating NAT. This results the hole punching fail and SLT cannot establish a direct connection. When the TTL value sets too high, the TCP SYN packet will reach the opposite NAT. If the Si filtering behavior of the opposite NAT is silent drop, the originating NAT needs to allow packet sequence “SYN-out, SYN-in” for establishing a direct connection. If the opposite NAT sends back a TCP RST packet, the originating NAT has to allow packet sequence “SYN-out, RST-in, SYN-in”.

P2PNAT

According to our observation, even both TCP SYN packets cross in the network, P2PNAT still needs one of the NATs to allow TCP packet sequence “SYN-out, SYN-in” for establishing a direct connection. If one of the TCP SYN packets arrives the opposite NAT before the other one leaves, the situation is similar to SNT. The originating NAT should allow packet sequence “SYN-out, SYN-in” when the Si filtering behavior of the opposite NAT is silent drop. If the opposite NAT sends back a TCP RST packet, the originating NAT should allow packet sequence “SYN-out, RST-in, SYN-in” for establishing a direct connection.

ESi

According to our observation, ESi needs at least one of the NATs to allow the incoming TCP SYN packet in ESi filtering. Although the DCR of ESi is the lowest, ESi can reuse the existing mapped-address. Besides, unlike other TCP NAT traversal schemes, both NATs in ESi generate zero error messages when establishing a direct connection successfully.

Use TCP NAT traversal schemes altogether

There are two ways of using all TCP NAT traversal schemes for establishing a direct connection to improve the DCR. One can use all schemes in parallel or sequentially. However, if we execute all schemes in parallel, at least 21 messages are generated during the experiment as shown in Table 2. This consumes more network resource than using just one scheme. As Table 2 illustrates, if one uses all schemes sequentially, in the worst case, the accumulative delay is much longer than either one of the schemes. According to our observation, the accumulated delay is almost four times longer than using only one scheme.

Priority in choosing a TCP NAT traversal scheme

When using all schemes sequentially, there should be a priority for each scheme. Although ESi has the lowest DCR, as long as one of the NATs allows incoming TCP SYN packet in ESi filtering, both hosts can establish a direct connection no matter how the opposite NAT behaves in Si filtering. As a result, ESi should have the highest priority. Although SNT has lower DCR than that of SLT and P2PNAT, TTL value is not a issue to SNT. So SNT gets higher priority compared to SLT and P2PNAT. Even though SLT and P2PNAT have the same DCR, P2PNAT has the limitation of simultaneous open. Thus, P2PNAT receives the lowest priority.

V. Conclusions

According to the experiment results, no single TCP NAT traversal scheme can achieve 100% DCR. Different TCP NAT traversal scheme has different features. Each scheme generates a specific packet sequence and the packet sequence has its own applicable NAT types. Only the NATs with such NAT types could establish a direct connection. Therefore, a TCP NAT traversal scheme is tightly coupled with NAT behaviors.

Brute force style of using TCP NAT traversal schemes may induce a long delay or excessive resource demand for setting up a connection. We observe that different NAT combinations have different applicable TCP NAT traversal schemes. When designing a TCP NAT traversal scheme, there should be a complete behavior test for both NATs. With information about NAT behavior, both hosts can choose a proper scheme for establishing a direct connection instead of using all schemes altogether.

Acknowledgements

This work was supported in part by National Science Council under Grant NSC 100-2221-E-009-072-MY3 and D-Link Co., Taiwan.

References

[1] P. Srisuresh and K. Egevang, Traditional IP Network Address Translator (Traditional NAT). IETF RFC 3022, 2001.

[2] B. Ford, P. Srisuresh, and D. Kegel. “State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs),” Proc. *USENIX Annual Tech*. Conf., 2005, pp. 179–92.

[3] P. Srisuresh, B. Ford, and D. Kegel. State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs). IETF RFC 5128, 2008.

[4] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. IETF RFC 5382, 2008.

[5] S. Guha and P. Francis, “Characterization and Measurement of TCP Traversal through NATs and Firewalls,” Proc. *Internet Measurement Conf*., 2005.

[6] J. Rosenberg et al. Session Traversal Utilities for NAT (STUN). IETF RFC 5389, 2008.

[7] F. Audet, Ed. and C. Jennings. Network Address Translation (NAT) Behavioral Requirem. IETF RFC 4787, 2007.