Extended UDP Multiple Hole Punching Method to Traverse Large Scale NATs

Kazuhiro Tobe 1, Akihiro Shimoda 1 and Shigeki Goto 1

1 Waseda University / 3-4-1 Okubo Shinjuku-ku Tokyo, Japan

E-Mails: {tobe, shimo, goto}@goto.info.waseda.ac.jp

Tel.: +81-3-5286-3182; Fax: +81-3-5286-3182

Abstract: A Network Address Translator (NAT) is a popular technological tool used in networks, especially in small-sized networks. Recently, network operators have been considering deploying Large Scale NATs (LSNs) to cope with IPv4 address pool exhaustion. This will make it necessary to deal with several problems related to LSNs, such as multiple levels of NATs (cascaded NATs) and the shortage of port numbers used by NATs. To address these issues, this paper extends the concept of UDP Multiple Hole Punching previously proposed by us. The use of our proposed method enables an accurate Port Prediction and reduces the number of open ports. The new method can determine the low TTL values for IP packets. We also discuss the application of i-Path routers, which provide status information about NATs along a network path for end hosts. The use of these routers makes it easier to perform NAT traversal.

Keywords: NAT; NAT Traversal; Large Scale NAT; UDP Hole Punching; P2P.

1. Introduction

A Network Address Translator (NAT) [19] is a popular technological tool used in networks, especially in small-sized networks. It is well known that some application software and tools cannot work properly with NATs by various reasons. There have been several approaches to solve this problem. They are called NAT Traversal methods.

Recently, network operators have been considering deploying Large Scale NATs (LSNs) [12] or Carrier Grade NATs (CGNs) to cope with IPv4 address pool exhaustion [9, 10]. An LSN can reduce the number of global IPv4 addresses needed. As of January 19, 2010, less than 10% of the total IPv4 address space was unassigned. The number had dropped to less than 8% by April 9 [25]. If it continues to follow the same trend, IPv4 address pool exhaustion will occur within two years [7]. Therefore, it is natural for a network operator to deploy LSNs or CGNs. However, the existing NAT Traversal methods cannot be simply scaled for LSNs or NGNs. It is necessary to deal with several problems when using LSNs or CGNs [4]. This paper discusses these issues, which include multiple levels of NATs (cascaded NATs) and the shortage of port numbers used by NATs.

We proposed a UDP Multiple Hole Punching method [21], which extends the original concept of UDP Hole Punching [6]. Our UDP Multiple Hole Punching method can be applied to Symmetric NATs [16] which cannot be easily handled by using plain NAT traversal methods. Our method predicts the next port number assigned to the host (Port Prediction). If the Port Prediction fails, a large number of ports are opened in order to traverse a Symmetric NAT. In our earlier method, the Time To Live (TTL) field had a low value in the IP packet header, such that the packet was discarded between a NAT in the sender side and the NAT in the destination side. It is important to determine an appropriate TTL value (Low TTL Value Determination), when the end hosts and servers do not possess the network path information.

This paper extends the concept of our earlier method for working with LSNs or CGNs. The new method can be applied to multiple levels of NATs (cascaded NATs). The new method improves the Port Prediction accuracy. It reduces the number of open ports based on the information. We also propose a simple method for determining the low TTL value. Our method can be used with i-Path routers to provide information about the NATs along the path. This information is utilized by the end hosts behind the NATs for successful NAT Traversal.
The rest of this paper is arranged as follows. In Section 2, we explain NAT. Section 3 describes NAT Traversal method. Section 4 provides details about the LSN or CGN. In Section 5, we propose our new method. Section 6 discusses the new method and Section 7 concludes the paper.

2. NAT Technology

It is possible to translate private IP addresses [13] into global IP addresses at the boundary between a local network and the Internet. This makes it possible for private local hosts to access the Internet. This address translation is called Network Address Translation (NAT) and a device to translate addresses is called a Network Address Translator (NAT) [19]. In addition to IP addresses, a Network Address and Port Translator (NAPT) also translates the port numbers of transport protocols (e.g., TCP or UDP). NAPT makes it possible for multiple hosts to share a single global IP address. Both NAT and NAPT are usually called NAT because most current broadband routers have the NAPT function.

2.1. Taxonomy of NATs

There are many types of NAT implementation. NATs are classified into four types in RFC 3489 [16]. NAT Traversal has the following order of difficulty: (easiest) Full Cone NAT < Restricted Cone NAT < Port Restricted Cone NAT < Symmetric NAT (most difficult). Most of the existing NAT Traversal methods cannot traverse a Symmetric NAT. These terms, i.e., Cone NAT (Full Cone NAT, Restricted Cone NAT, and Port Restricted Cone NAT) and Symmetric NAT, are traditionally used in the literature for NAT Traversal. Therefore, they will be used in this paper. It has been said that the terms and classifying algorithms used in RFC 3489 are inadequate to describe the behavior of a NAT [15]. RFC 4787 [1] explains the behaviors of NATs instead of terms such as Cone NAT and Symmetric NAT. RFC 4787 also describes many characteristics. We will refer to two of these features in this paper: (1) Address and Mapping Behavior and (2) Mapping Refresh. These features are closely related to our proposed method.

2.1.1. Address and Mapping Behavior

When the host behind a NAT establishes multiple sessions with a different external host, the NAT allocates a new endpoint <IP address, port number> or reuses the mapping created in the previous session based on the implementation of the NAT. This behavior is called the Address and Mapping Behavior. RFC 4787 classifies this behavior into three groups: (1) Endpoint-Independent Mapping, (2) Address-Dependent Mapping, and (3) Address and Port-Dependent Mapping.

(1) Endpoint-Independent Mapping

Endpoint-Independent Mapping NAT (EIM-NAT) allocates the same endpoint <AX, PN> whenever a local host (Host-L) <AX, PL> sends a packet to any external endpoint <any, any>. Figure 1 illustrates the situation. EIM-NAT is called Cone NAT in RFC 3489.

(2) Address-Dependent Mapping

Address-Dependent Mapping NAT (ADM-NAT) allocates a new endpoint <AX, PN’> when a local host (Host-L) <AX, PL> sends a packet to an external host <AX’, any> (AX is not equal to AX’) to which Host-L has not sent a packet yet. That is, ADM-NAT uses the same endpoint for those packets whose destination IP address is the same. However, it assigns a different endpoint to packets whose IP address is different from the previous ones. Figure 2 shows this mapping. In UDP Hole Punching (described later in subsection 3.1), it is necessary to predict the new port number (PN’). Both ADM-NAT and APDM-NAT (described below in (3)) are called Symmetric NAT in RFC 3489.

(3) Address and Port-Dependent Mapping

Address and Port-Dependent Mapping NAT (APDM-NAT) maps a new endpoint when a local host (Host-L) <AX, PL> sends a packet to an external endpoint to which Host-L has not sent a packet yet. That is, APDM-NAT assigns a new endpoint to the packets if either the destination IP address or the destination port number is different from previous ones. Figure 3 explains the new endpoints. A new endpoint <AX, PN’> is mapped to the packets sent to endpoint <AX, PN’>. A new endpoint <AX, PN’’> is mapped to the packets sent to endpoint <AX, PN’>. In UDP Hole Punching (described in subsection 3.1), it is necessary to predict this new port (PN’). Both ADM-NAT and APDM-NAT (described in (2)) are called Symmetric NAT in RFC 3489.
2.1.2. Mapping Refresh
A TCP connection is initiated by a 3-way handshake (SYN, SYN/ACK, and ACK) and terminated by FIN and ACK packets. The initiation packets and the terminations packets in TCP connections are well defined. A NAT registers a new mapping entry when it observes the start of a TCP connection and deletes mapping entry when it observes the end of the TCP connection. UDP sessions have neither a well-defined initiation nor termination because UDP is a connection-less protocol. Therefore, a NAT uses a timer to maintain a record of mappings. This timer is updated when a UDP packet related to a mapping is observed. When the time is over, the mapping entry is deleted from the NAT table. In that case, UDP Hole Punching [6] (described in subsection 3.1) must be initiated again. In UDP Hole Punching, a host needs to send keep-alive packets at appropriate time intervals. The time-over parameters are different from NAT to NAT. It is difficult to determine the appropriate time interval. The only method for doing so is heuristic learning through trial and error. Section 5.3.2 proposes a method to solve this problem.

3. NAT Traversal
In general, external hosts (hosts outside of NATs) cannot connect to internal hosts behind NATs. This means Peer-to-Peer (P2P) communications cannot work between two hosts behind different NATs [18]. Users cannot enjoy some types of online games and Voice over IP (VoIP) applications, e.g. IP telephone or TV conference systems, on hosts behind NATs. To solve this problem, NAT traversal techniques have been developed.

3.1. UDP Hole Punching
UDP Hole Punching [6] is a NAT Traversal method. Figure 4 shows a sequence diagram for UDP Hole Punching. The method has three steps. (1) First, host-A sends a UDP packet to host-B. That is, the packet punches a hole. (2) Second, host-B sends a UDP packet toward the open port on host-A. This packet reaches host-A because it is the return packet of the first packet. (3) Third, host-A returns a UDP packet to host-B. At this time, the packet reaches host-B because this is the return packet of the second one. Then, a UDP session begins between the two hosts behind different NATs, NAT-A and NAT-B.

![Figure 4. Sequence diagram of UDP Hole Punching](image)

In UDP Hole Punching, it is necessary to know each host's external endpoint <external IP address, external port number> assigned by the NAT, instead of the host's internal endpoint <host's IP address, host's port number>. Usually, this information is obtained from a rendezvous server, which has a global IP address on the Internet. Host-A and host-B can communicate with the rendezvous server even before Hole Punching is established. This earlier communication is called step (0). The rendezvous server checks the packets and obtains the hosts' external endpoints. Then, the rendezvous server informs host-A and host-B of the endpoint information. However, Symmetric NATs [16] assign a new port number if the destination endpoint is different. Therefore, Symmetric NATs assign port numbers at step (0) that are different from those at step (1). Consequently, the NATs discard packets in steps (2) and (3). In fact, the next port number assigned by the Symmetric NAT algorithm is sometimes predictable due to the regularity of the port assignment algorithm. We can utilize this prediction.

3.2. UDP Multiple Hole Punching
As described above, UDP Hole Punching cannot traverse Symmetric NATs in general. However, the next port number assigned by a Symmetric NAT is sometimes predictable due to the regularity of the port assignment algorithm. UDP Multiple Hole Punching [22] can traverse Symmetric NATs without relay servers (cf. Interactive Connectivity Establishment (ICE) [14]). UDP Multiple Hole Punching is friendly with real-time applications such as online games and VoIP applications because it does not relay packets. Relaying packets leads to heavy loads on servers and introduces extra delay time. In UDP Multiple Hole Punching, a host communicates with two servers to obtain the information needed for predicting the regularity of the port assignment algorithm. Based on this information, hosts or servers can predict the next port number assigned by the NAT. This technique is called Port Prediction. Unfortunately, if Port Prediction fails, as a last resort UDP Multiple Hole Punching hosts send a large number of packets with low Time To Live (TTL) value in order to open numerous ports. Although this increases the success rate for traversing Symmetric NAT, it is wasteful because the port numbers are limited resources.

4. Issues in LSN or CGN
Recently, network operators have been considering deploying Large Scale NATs (LSNs) [12] or Carrier Grade NATs (CGNs) to cope with IPv4 address pool exhaustion [9, 10]. An LSN can reduce the number of global IPv4 addresses needed. As of January 19, 2010, less than 10% of the total IPv4 address space was unassigned, and this had fallen to less than 8% by April 9 [25]. If it continues to follow the same trend, IPv4 address pool exhaustion will occur in two years [7]. However, the existing NAT Traversal methods cannot be simply scaled for LSNs and NGNs. It is necessary to deal with several problems to utilize LSNs or CGNs [4].

4.1. Port Number Limitation
With LSNs or CGNs, there is a limitation on port numbers available for each user sharing an IP address. Thus, LSNs may restrict some types of applications. They may block applications that accept accesses from the Internet at a specific port number or applications that use numerous port numbers to establish multiple sessions. It is well known that Google Maps establishes multiple sessions. Each session downloads a part of a map and draws on the screen concurrently with the others. If some sessions are not established, the maps drawn on the screen will appear to be worm-eaten [10]. A survey [10] showed that iTunes establishes 230 to 270 sessions, and Amazon.com and YouTube establish 90 sessions concurrently. These applications have the same problem.
as Google Maps. A typical Windows PC always establishes at least 5 to 10 sessions because of update processes such as Windows Update and Anti-virus software, even if no other applications are running.

NATBLASTER [2] is a TCP version of UDP Multiple Hole Punching (i.e., NATBLASTER is a TCP Hole Punching method for traversing a Symmetric NAT). It shows Hole Punching success rate in a case where a host sends \( n \) packets (the destination port number is changed packet by packet) to a host that opens \( n \) ports behind a Symmetric NAT. According to NATBLASTER, Hole Punching has a success rate of \( p(n) \) if it is not possible to predict the port number assigned by the Symmetric NAT. \( p(n) \) is calculated as follows, where \( N \) refers to the number of possible port choices (\( N = 65535-1023 = 64512 \) if the NAT can assign any ports except the well-known port numbers from 1 to 1023).

\[
p(n) = 1 - \prod_{i=0}^{n-1} \frac{N - i - n}{N - i}
\]

For example, it is necessary for a receiver to wait with 439 ports open and for a sender to send 439 packets (\( n = 439 \)) in order to obtain a success rate of 95\% (\( p(n) = 0.95 \)). The value 439 is a large number of ports for hosts restricted by LSNs. The number of ports available for users may be insufficient to apply NATBLASTER. Even if there are enough ports, NATBLASTER wastes the precious resource of port numbers restricted by LSNs.

The same is equally true of UDP Multiple Hole Punching. Therefore, when using UDP Multiple Hole Punching behind LSNs, it is important to improve the Port Prediction accuracy in order to reduce the number of open ports. In particular, the Port Prediction should not fail, because taking the last resort (opening numerous ports) wastes numerous port numbers. With the naive UDP Multiple Hole Punching [21], there is a possibility of false positives (mistaking a predictable port assignment NAT for a random one). This is because packets may be unfortunately sent from other irrelevant hosts, and then the Symmetric NAT assigns a new port number for the new session. This may disturb the Port Prediction. This error leads to the last resort and wastes port numbers. If this happens, it significantly decreases the success rate of the UDP Multiple Hole Punching in an environment where the port numbers available for users are limited by LSNs. In subsection 5.1, we propose new methods that take other hosts into consideration to increase the Port Prediction success rate.

### 4.2. Multiple levels of NATs

When UDP Multiple Hole Punching hosts open numerous ports, the hosts send UDP packets whose TTL is set so low that the packets are dropped between the NAT on the sender side and the NAT on the destination side. For example, a host has to send a packet whose TTL is set between 1 and 5 in the network illustrated in Figure 5. However, it is difficult to determine an appropriate TTL value (Low TTL Value Determination) because the end hosts and servers do not have information about the NATs along the path. Therefore, Yuan Wei, the inventor of UDP Multiple Hole Punching, determined TTL values in accordance with an experimental network environment [21]. NATBLASTER [2] mentions using a method like Traceroute (i.e., increasing TTL by 1) but this is not practical. To make matters worse, deploying LSNs makes the path between end hosts complex (drawn in Figure 6) and Low TTL Value Determination difficult. This is because end-hosts cannot know how many NATs are cascaded along the path. In subsection 5.2, we propose a considerably simpler method for estimating an appropriate TTL.

![Figure 5. Network between hosts behind NATs](image)

![Figure 6. Complex network between hosts behind multiple levels of NATs](image)

### 5. Proposed Method

This paper proposes a new method for improving UDP Multiple Hole Punching. It is based on two techniques. The first technique extends the concept of the Port Prediction method to improve the accuracy. Another technique determines a low Time to Live (TTL) appropriately and simply.

#### 5.1. Extended Port Prediction

UDP Multiple Hole Punching has the problem that hosts may open more ports than necessary. This is because it does not take into account the Port Prediction error caused by Symmetric NATs [16]. Symmetric NATs may assign new port numbers for other hosts, which start new connections during the Port Prediction. If this type of assignment occurs, the port assignment algorithm of the NAT is estimated to be random, while it is really a predictable one. In order to solve this problem, we propose two methods for improving the Port Prediction accuracy. These methods take the packets sent from other hosts into consideration. The first method is the Capturing Method (described in section 5.1.1) and the second is the Scanning Method (described in section 5.1.2).

In UDP Multiple Hole Punching, servers tell the hosts the next port number (e.g., 12345) predicted to be assigned by the Symmetric NAT. We extend the UDP Multiple Hole Punching method to give additional information to the hosts. This new information is the range of error (e.g., \([0, 5]\)), which is predicted by the Capturing Method or Scanning Method. The range of port numbers that can be assigned to the next packet by a NAT is calculated by combining the information (e.g., \(12345 + [0, 5] = [12345, 12350]\)). Then, a receiver opens these ports and a sender sends packets whose destination port numbers are these port numbers. If a Symmetric NAT assigns one of these ports, the hosts do not have to send numerous packets and open numerous ports. They only have to send a few packets and open a few ports. This is how this extension decreases the unfortunate possibility of hosts opening numerous ports. It also decreases the number of
open ports, even if the hosts and servers fail to find the regularity of the port assignment by a Symmetric NAT.

5.1.1. Capturing Method
In the Capturing Method, the newly extended method captures packets in the network behind NATs in order to observe outgoing UDP packets during Port Prediction. The observer counts the number of packets from hosts other than the target of prediction to the endpoint which has never been observed. These packets may be assigned a new port number by the Symmetric NAT, which disturbs the Port Prediction. It should be noted here that the observed packets may or may not be the initiation packets of a UDP session because the new method observes the network traffic only during the Port Prediction. The initiation packets are not clear in UDP sessions, unlike in TCP connections, which are established by a 3-way handshake (SYN, SYN/ACK, and ACK). The initiation packets of a UDP session may be sent before the Port Prediction.

The number of initiation packets to which Symmetric NATs assign new port numbers is less than or equal to the number of outgoing UDP packets from other hosts to new endpoints observed during the Port Prediction. It is necessary to take this into consideration when applying the new method. It is not possible to know the exact number of newly assigned mappings during the Port Prediction. That is, the Capturing Method does not estimate the exact number but rather the range of numbers in Port Prediction. Nevertheless, the Capturing Method works to some extent because it is important to decrease the number of open ports in the port-restricted networks behind LSNs.

5.1.2. Scanning Method
In the Scanning Method, the new method counts how many hosts are working in the network segment before the Port Prediction. For example, a UDP Multiple Hole Punching host whose IP address is 192.168.0.2 (a typical Class-C private IPv4 address [13]) and whose subnet mask is 255.255.255.0 (i.e., 192.168.0.2/24) can send Ping to every IP address between 192.168.0.1 and 192.168.0.254. It is also possible to send an Address Resolution Protocol (ARP) request or some UDP packets, or establish TCP connections instead of sending Ping. Any method is fine if it can examine whether or not each host in the network is alive. Then, the new method estimates the potential error ([0, E]) in the Port Prediction based on the number of hosts (N) detected by the scanning method. E is estimated from the following equation: E = w * N, where w refers to a weight affected by the time it takes the Port Prediction and so on. That is, the Scanning Method does not estimate the exact assignment of a new port but determines the range of error [0, E] in the Port Prediction.

5.2. Simple Method for Low TTL Value Determination behind Multiple Levels of NATs
As mentioned above, UDP Multiple Hole Punching hosts send UDP packets whose TTL value is set so small that the packets are discarded between the NAT on the sender side and the NAT on the destination side. It is difficult to determine an appropriate low TTL value when there is little information about the network configuration. The existing methods for determining TTL values are not practical. To make matters worse, LSNs of ISPs or NATs in houses and small offices make the network topology more complex and Low TTL Value Determination more difficult. It is impossible to know how many routers and NATs are cascaded in a network. It has been proposed that Traceroute or Tracert be used to obtain the IP addresses of routers and NATs along the path. If a node has a private IP address, the node is thought to be behind a NAT. However, hosts behind NATs can be assigned global IP addresses instead of private IP addresses. Moreover, it is known that some routers do not return ICMP messages for security reasons. Therefore, it is not practical to estimate an appropriate TTL value by looking at IP addresses from Traceroute or Tracert.

We propose a simple new method for Low TTL Value Determination in UDP Multiple Hole Punching. First, this method measures the hop count to the destination host by Traceroute or Tracert. Then, it sets the TTL value to half of the measured hop count. For example, if the hop count between a sender and a destination host is 12, the initial TTL value is set to 6 (= 12/2). This proposed method is based on the assumption that NATs are concentrated close to end hosts and do not exist in the center part of a network even if LSNs of ISPs or NATs at houses or small offices make the network topology complex (drawn in Figure 6).

The proposed method requires only the hop count to the destination. It does not matter whether the routers along the path are configured not to return ICMP messages (Ping). Tracert sends ICMP packets.

5.3. NAT Traversal by i-Path Network Transparency
The proposed method can be used with i-Path routers which realize network transparency.

5.3.1. i-Path Routers
i-Path [8, 11, 24] is a new framework for end-hosts to access network status information along a path. i-Path routers also provide end hosts with information about the path. It is possible to combine them. i-Path takes in-band cross layer approaches. i-Path makes it possible for end hosts to obtain information about the network status from the routers along a path. If we use i-Path routers, the end hosts can obtain various information, e.g., geographical location, network throughput, and traffic volume. i-Path observes the information disclosure policy of routers. It discloses only the information that all of the stakeholders (i.e., the sender, receiver, and ISPs along the path) allow to be disclosed.

5.3.2. Disclosing Information about the NATs
The proposed method estimates information about the NATs (e.g., the algorithms for the port assignment and the timer used to maintain the mappings) by sending some packets and examining the behavior by the UDP Hole Punching and UDP Multiple Hole Punching. There are no problems if the information is provided as i-Path information. Disclosing information about whether the NAT function is on or off on a router is also useful. As already mentioned, examining these data is sometimes troublesome and the results are not always accurate. Therefore, we propose using i-Path routers to disclose the information about NATs [20]. Figure 7 shows an example of this disclosure. The proposed method can save the time and resources that are necessary for Hole Punching.

![Figure 7. Transparent network by i-Path](image-url)
This combined method can also improve the reliability of the information. The method proposed in subsection 5.1 and 5.2 depends on the estimation, but this combined method is based on the accurate information provided by i-Path routers.

5.4. Evaluation
We implemented several Java programs in order to evaluate the proposed method. Unfortunately, Java does not have an API to set an initial TTL value. Therefore, Java programs invoke a Ruby program, which sends a specified number of UDP packets with a specified TTL value via the java.lang.Runtime.exec() method.

Figure 8 shows the testbed which, consists of five networks: two home networks, two ISP networks, and a global network. It was constructed using VMware ESXi, which offers virtual networks with multiple virtual machines and virtual switches on a single physical machine. Virtual switches configured to promiscuous mode can be used for virtual repeater hubs. Virtual routers can be realized by running virtual machines as routers. We adopt the Flexible NAT Emulation Server (flexNES) because Iptables cannot emulate a Symmetric NAT. The characteristics of a NAT (e.g., Address and Port Mapping Behavior, Mapping Refresh, and so on) are configurable on a flexNES.

6. Discussion
6.1. Capturing Method vs. Scanning Method
The Scanning Method estimates the range of error stochastically on the basis of the number of hosts in the network. In contrast, the Capturing Method estimates it by observing the packets in the network. Therefore, the Capturing Method yields a greater improvement in the Port Prediction accuracy than the Scanning Method. However, the Capturing Method requires root authority or Administrator authority to capture packets. On the other hand, the Scanning Method only requires user authority if it uses Ping or TCP/UDP packets. As stated above, the Capturing Method and Scanning Method both have merits and demerits. Therefore, it is better to use both methods as the situation demands.

6.2. Scope of Application
In this paper, we assume that LSNs are deployed specifically using the NAT-444 model [17], which causes multiple levels of NATs (cascaded NATs). In fact, there are several models that are used to deploy LSNs, and some models do not take multiple levels of NATs. However, all of these models, including the Dual-stack lite (DS-lite) [5] and Address plus Port (A+P) [3] models used for deploying LSNs with tunnels, have the same port number limitation as the NAT-444 model. Therefore, the application area of our proposed method is not limited to the NAT-444 model, but includes any other models for deploying LSNs. In addition, some users are behind multiple levels of NATs because the NATs in their home or building are actually cascaded. Our newly proposed method has a wide area of application.

6.3. Comparison with UPnP
Universal Plug and Play (UPnP) [26] is a popular protocol that allows PCs, information appliances, and wireless devices to connect with home networks seamlessly. UPnP Internet Gateway Device (IGD) compatible NAT routers can be configured by hosts behind NATs. These hosts can obtain port mapping information and configure port forwarding. However, such hosts can access a UPnP IGD only in the local network. This means UPnP does not work in networks that are behind multiple levels of NATs. Furthermore, UPnP does not have an authentication mechanism.

Our proposed method, Extended UDP Multiple Hole Punching and NAT Traversal by i-Path Network Transparency, is quite effective against multiple levels of NATs. Furthermore, the disclosing policy is flexibly configurable in i-Path. Our proposed method therefore has an advantage over UPnP.

6.4. Comparison with End-to-End NAT
End-to-End NAT [12] advertises the existence and state of the NAT and the end hosts complement the NAT behavior to achieve end-to-end transparency. Our newly proposed method by i-Path discloses NAT information to complement NAT Traversal. While End-to-End NAT requires that end hosts have the OS kernel fixed, but our proposed method does not have to require such fixes.

6.5. Future Work
The specific Low TTL Value Determination proposed in subsection 5.2 is based on the assumption that the NATs are concentrated near the end hosts and do not exist near the center of the path (as shown in Figure 5) if LSNs of ISPs or NATs in houses or small offices make networks complex. The verification of this assumption will be performed in our future work.

7. Conclusion
ISPs have been planning to deploy LSNs in order to save global IP addresses, whose pool will be exhausted in the near future. This will create some problems such as multiple levels of NATs (cascaded NATs) and port number limitations for certain types of applications. The existing NAT Traversal methods are not capable of addressing these issues. We have proposed UDP Multiple Hole Punching, which is a NAT Traversal method for traversing Symmetric NATs effectively. However, it is necessary to improve the Port Prediction accuracy in order to reduce the number of ports, which is restricted by LSNs. In addition, we need a new Low TTL Value Determination method that can accommodate new network configuration with multiple levels of NATs.

This paper proposed an extension of the UDP Multiple Hole Punching method to address these issues. The Capturing Method and the Scanning Method improve the Port Prediction accuracy. This decreases the unfortunate possibility of hosts opening numerous ports and also decreases the number of open ports, even if hosts and servers fail to find the regularity of the port assignment by a Symmetric NAT. Our proposed Low TTL Value Determination method is simple but practical in a network where NATs are cascaded. The disclosure of the NAT information by i-Path routers makes it possible for end hosts to obtain accurate information from routers along the path. It has been difficult for
existing methods to accurately guess the timer values to maintain the mapping and port assignment algorithm by a Symmetric NAT. Our new method solves these problems.

Acknowledgments
This project has been supported by the National Institute of Information and Communications Technology (NICT), Japan.

References
[25] IANA IPv4 Address Space Registry, http://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xml