The Journal of Systems and Software 82 (2009) 1620–1626

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Challenge and solutions of NAT traversal for ubiquitous and pervasive applications on the Internet

Yaw-Chung Chen [a,*], Wen-Kang Jia [b]

[a] Department of Computer Science and Information Engineering, Asia University, 500, Liufeng Road, Wufeng, Taichung, Taiwan, ROC
[b] Department of Computer Science, National Chiao-Tung University, 1001, Tahsueh Road, Hsinchu, Taiwan, ROC

ARTICLE INFO

Article history:
Available online 5 April 2009

Keywords:
NAT (Network Address Translator)
Ubiquitous
Pervasive
P2P(Peer to Peer)
NATng (NAT next generation)

ABSTRACT

Network Address Translator (NAT) has brought up many changes and opportunities to the Internet. How do the ubiquitous and pervasive applications coexist with NAT and interoperate with each other? In this article, we discuss the essence of NAT sensitive applications as well as the challenge and response for various NAT traversal solutions. All questions pointed to redesign a new NAT framework with a major change to accommodate NAT problems all at once. We introduce a novel next generation NAT (NATng) framework, which consists of a Bi-directional NAT (BNAT) and a Domain Name System Application Level Gateway (DNS_ALG) with a Border Network Address Translator Control Protocol (BNATCP) function to control all BNATs. The above components coordinate and provide bidirectional access capability between intranet and Internet, so all private hosts can be addressed via Fully Qualified Domain Name (FQDN). Logistically, NATng extends the IPv4 address space from 2^{32} to 2^{48} or even more. It features high potential to solve the problems for ubiquitous and pervasive applications which may encounter IPv4 address exhaustion on the current Internet.

1. Introduction

Ubiquitous computing, often synonymously called pervasive computing, mobile computing and etc. is an emerging field of research that brings up revolutionary paradigms for computing models in the 21st century (Weiser, 1991). There is tremendous volume of developments in related technologies such as wireless communications and networking, mobile computing and handheld devices, embedded systems, wearable computers, sensors, RFID tags, smart spaces, middleware, software agents, and the like (Satyanarayanan, 2001).

The goal of ubiquitous computing is to create ambient intelligence where network devices embedded in the environment provide unobtrusive connectivity and services all the time and everywhere, thus improving human experience and quality of life without explicit awareness of the underlying communications and computing technologies. In this environment, the world surrounding us is interconnected as pervasive network of intelligent devices which cooperatively and autonomously collect, process and transfer information, in order to adapt to the associated context and activity (Saha and Mukherjee, 2003). Such systems are now affecting every aspect of our life to the point that they are hidden inside various appliances.

IP network is used to connect all the end hosts that can exchange or share information among each other in the Internet nowadays. The current IP version 4 (IPv4) is based on a 32-bit address for all endpoints since 1970s. Those endpoints can be computers, printers, routers, exchangers, gateways or other networking devices whose location can be uniquely identified (Srisuresh et al., 1999). Due to the dramatic growth of internet population in the past two decades, the originally designed IPv4 address space would be exhausted in the foreseeable future. This IP address shortage problem usually causes Internet users unable to accommodate all endpoints trying to access the Internet. To resolve this issue, a simple way is using an IP address-sharing gateway that separates the public Internet and the private network (Rekhter et al., 1996). This kind of gateway devices translates the private IP address to global IP address and vice versa based on the NAT mechanism (Aboba and Dixon, 2004).

The first challenge of agent-oriented ubiquitous and pervasive applications is that they need mass IP address space on the Internet. This surpasses the capacity of IPv4. Second, the deployment of NATs may slow down the above IP shortage issue, but it encounters a side-effect that is addressed in the next section. Third, due to the booming of broadband technologies and media streaming based ubiquitous applications nowadays, the NATs used currently may become the performance bottleneck of Internet access. The NAT's problems are not merely as mentioned above, in this article our discussions focus on the addressing in the NAT enabled network

* Corresponding author.
E-mail address: ycchen@us.nctu.edu.tw (Y.-C. Chen).

0164-1212/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2009.03.078

environment as well as on solving the side effect brought up by NATs (Srisuresh and Egevang, 2001).

Although the next generation Internet protocol IPv6 provides 128-bit address space, it may take many years for Internet Service Provider (ISP) to upgrade their network equipments such as routers, and Internet users to update their endpoint software such as operating systems. There is no D-day that the Internet world will suddenly change to IPv6 overnight, and NAT is still the essential technique for extending the life cycle of IPv4.

2. Traditional NAT and its problems

NAT is just a generic acronym. It also includes the extension of Network Address Port Translator (NAPT) (Egevang and Francis, 1994). The principle of using traditional NAPT to translate the IP packet header is as follows: an IP packet carries either TCP or UDP segment, which contains a 16-bit source/destination port number for the identification of services. Using this port number as the extension of IP address, the network address translator can provide the function of IP sharing. It translates multiple private IP addresses used in the internal network to a unique global (or public) IP address, and uses the source port number to differentiate these packets from different endpoints (Srisuresh and Holdrege, 1999). In the translation process, the NAT will generate the translation table adaptively. It not only applies a standard translation procedure to the outgoing packets but also uses the inverse translation procedure for the incoming packets. Fig. 1 shows the packet flow of NAT process. It is essential to replace the internal private network IP address with the external public network IP address. Here the usage of port number is different from the idea in the original design of IP protocol; it keeps using the original port number in advance and replaces it when a conflict occurs, but in certain designs it replaces the port number with new ones without exception. The checksum field of the TCP/UDP header always needs recalculation (Aboba and Dixon, 2004).

Applications with certain characteristics may have "**NAT-Sensitive**" properties as follows (Senie, 2002):

(1) **Protocols not supported**: The network and/or transport layer protocols used by applications are blocked by NATs natively. These protocols include IPv6, SCTP (Stream Control Transmission Protocol), and DCCP (Datagram Congestion Control Protocol) etc. which were developed after the deployment of NATs (Holdrege and Srisuresh, 2001).

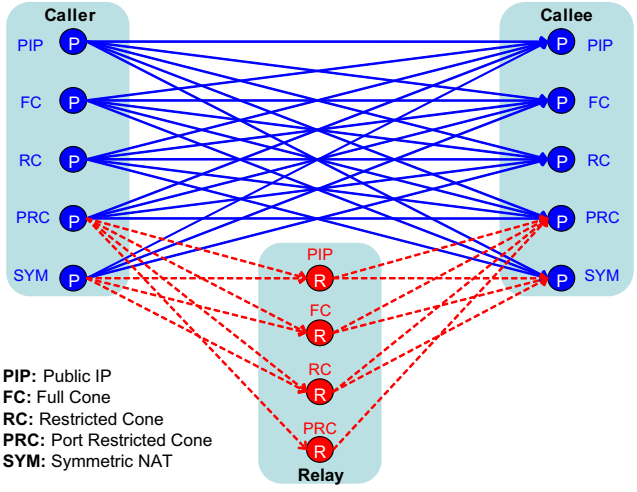

Fig. 1. The connection patterns between nodes behind various NAT types.

(2) **High security**: The applications disallow messages spoofed during transmission. For instance, IPSec recognizes the IP header modified by NATs as a security alert because IPSec cannot work together with NATs.

(3) **Establishment latency**: The applications are concerning real-time property during their connection establishment. Even though NAT traversal mechanism exists, the connections will still be timeout due to the extended establishment latency. This kind of protocols includes VoIP, IPTV, etc.

(4) **Connection direction**: The applications have "Reverse Connection" or "Symmetric Connection" behavior. The so-called reverse connection means that the client is outside NATs and the server is inside NATs. In such case, a connection may be initiated from outside NATs. We know that the connection will be blocked by NATs (Srisuresh et al., 2002). As for symmetric connection, it means that application consists of multiple protocols and/or service ports in both source and destination. The backward connections from outside to inside will be blocked by NATs. It is expected that all Peer to Peer (P2P) applications have this property (Johnston and Sinnreich, 2006), so do FTP port mode (port 20 + 21), IP tunnel and VoIP (SIP + RTP) applications (Quittek et al., 2006).

The software architecture should follow some rules to prevent a program from becoming NAT-sensitive application mentioned above. Unfortunately, ubiquitous and pervasive applications often contain most of these characteristics. For example, in the sensor network architecture, most sensor-sides usually locate behind NATs, and the management-side needs to collect data from Internet periodically; we know that the sensor could be sleeping to save power, that implies the sensor cannot keep the binding records alive under NAT. Management must be done to establish a reverse connection from outside NAT for waking up the sensor. This is impossible in traditional NAT environment unless the static map has been setup.

However, we have static map setup for IP/port in NATs. It does not save the IP address, and/or increase the complexity of network management. In addition, most ubiquitous and pervasive applications have played the roles of content providers, and/or their cooperation behaves like a peer-to-peer model: every peer works as both a client and a server simultaneously.

3. Current NAT traversal solutions

Given the complexity of today's NAT and firewall-protected networks in residential, campus and enterprise environments, ubiquitous and pervasive applications can only be successfully provided with certain help from NAT traversal mechanism, which must therefore be able to solve NAT and firewall problems remotely. Since NAT problems have been pointed out, many NAT traversal solutions were proposed to recognize and traverse NAT devices which are manufactured by various vendors. Now we have STUN (Simple Traversal of User Datagram Protocol through Network Address Translators) (Rosenberg et al., 2003), TURN (Traversal Using Relay NAT) (Rosenberg et al., 2009), RSIP (Realm Specific IP) (Borella et al., 2001a,b), MIDCOM (Middle box Communication) (Swale et al., 2002), UPnP (Universal Plug and Play), ICE (Interactive Connectivity Establishment) (Rosenberg, 2003), Path-coupled signaling (Martin et al., 2005), etc. All of them are attempting to solve NAT problem in various perspectives, but each can reasonably be deployed only in certain situations. In other words, those are only partial solutions. Practically, for ubiquitous and pervasive applications, it is necessary to locate a middle box for relaying service traffic between nodes which are behind NATs. Moreover, the relay capacity also decides how much traffic intensity can be accommodated in the ubiquitous and pervasive systems.

3.1. Relay server

Generally, a relay server is reachable through predefined ports and protocols by any peer-node, and most likely owns a public IP address. If some peer-nodes behind symmetric NATs and/or port restricted cone NATs (Srisuresh et al., 2008) could not make direct connection to each other, a node will relay their traffic partially or totally. All connections through relay servers are same as that directly between nodes. Today, most mobile Internet access providers offer network services by WAP (Wireless Application Protocol), GPRS (General Packet Radio Service), EDGE (Enhanced Data Rates for GSM Evolution) and HSDPA (High Speed Downlink Packet Access) technologies; the mobile terminals are usually assigned a private IP addresses via dynamical allocation (behind NAT). Therefore unless working in an unordinary situation, developers of ubiquitous and pervasive applications and middle-ware should always assume that their systems will run on devices that are behind NATs. This causes ubiquitous and pervasive clients to highly rely on relay service for communication.

The major reason to use relay server is for NAT traversal. A ubiquitous and pervasive system could utilize a subset of nodes with higher capabilities such as sufficient network bandwidth, high computing power, low processing load, and large storage space, to enhance the quality and/or the functionalities of the service provisioning. These special nodes are often referred to as "relay-nodes", which can provide high bandwidth Internet connection and transparent TCP/UDP protocol utility. If the relay-node is far from the communicating peers, it may impair the service quality (especially latency). Occasionally, QoS through direct connection between node A and node B may be poorer than that through indirect connection via peer C. Study in (Ren et al., 2006) addressed that about 1–10% of connections with one-hop relay path experience shorter Round-Trip Time (RTT) than those connections using direct routing mode in the connecting sessions. Therefore, it is important to design a method for searching a suitable relay-node that is close to the end-peers behind NATs, so that unnecessary relaying delay can be reduced.

Sometimes relay-nodes (also a normal user) may leave the ubiquitous and pervasive system frequently. A connecting session may encounter leave or removal of its relay-node in the system, in that case the connecting session will be blocked suddenly, and it must perform relay-node handoff procedure; the probability of handoff depends on the percentage of average connecting time, average survival time of relay-nodes, number of peer in the system, etc. However, service through a relay node may be trade off by QoS degradation. Since frequent relay-node-handoff causes the handoff-dropping ratio to increase, we suggest avoiding relay node whenever possible because of the unpredictable behaviour of the single relay-node even it is a fixed server facility.

However, relay server is the only solution with 100% efficacy including TCP for NAT traversal. And its implementation for application deployment is relatively simple. The disadvantage of relay method may be the cost effectiveness and poor QoS (Rosenberg et al., 2009).

3.2. STUN

STUN is a protocol for assisting application to find out its public IP address/port and the type of NAT service it is sitting behind. STUN clients are entities that generate STUN requests. A STUN server is generally attached to the public Internet; they receive STUN requests and send STUN responses. Finally, STUN clients can recognize various types of NAT where they reside behind, such as (1) Full Cone; (2) Restricted Cone; (3) Port Restricted Cone and (4) Symmetric; then clients outside NATs could send UDP packet through NATs to reach the STUN client in specified IP address/port which is predefined in NATs by STUN servers.

The first major achievement by STUN is that it makes a simple classification of NAT types and an interoperating model in UDP behaviour (Fig. 1) (Jennings, 2007). Secondly, STUN establishes a standard test and traversal procedure to bind an accessible IP address/port (aka "hole-punching") to outside NATs (Rosenberg et al., 2008). STUN is near perfect in UDP applications, its only drawback is the effectiveness of TCP applications due to the disorderliness in the TCP protocol stack implementation of the NATs (Guha et al., 2008). If the application is based on UDP (Audet and Jennings, 2007), STUN should be sufficiently assisted with relay services (Rosenberg et al., 2009).

3.3. ICE

Further, "Interactive Connectivity Establishment (ICE)" (Rosenberg, 2003) is a novel method that tries to build NAT traversal intelligence into nodes so that they can perform route discovery, relay lookup, path optimization, and even verify media flow before a connection is deemed to be established. Prior to sending a request, the caller executes a sequence of steps to characterize the type of NAT with which it is associated. First, a caller obtains addresses of all available interfaces; then it checks the results of those reachable peers from STUN server; sometimes a caller peer can't find a direct data path to the corresponding node so it needs to negotiate a usable port with the relay server(s).

Through ICE procedure, a predictable smaller percentage of nodes behind NATs really need the relay server with public IP address, or behind NATs which belong to full cone, restricted cone and port restricted cone, but not behind symmetric NATs; and the nodes can be using public IP or behind any NAT types. Their relationship is shown in Fig. 1. In general, relay servers behind NATs are inconvenient – before nodes use a relay server which is behind NAT (excluding symmetric NATs), they must perform ICE procedure between both sides first. If they selected a relay server behind port restricted cone NATs, the relay service for one side nodes behind symmetric NAT would not work. If they selected a relay server behind symmetric NATs, any relay service for nodes behind port restricted cone and symmetric NATs would not work. Therefore, a simple system usually does not select a host behind NATs as a relay server (Boulton et al., 2008).

ICE is becoming increasingly important for P2P-based ubiquitous and pervasive systems on the open Internet, as it enables NAT-bound nodes to provide accessible services. ICE provides best-effort direct connection between peers, and it can help peers discover qualified candidates for relay servers; that is, the STUN enabled nodes and servers to provide hole-punching and relaying services, respectively. An ICE (STUN) service deployed in super-nodes or fixed servers is suggested. A super-node may instruct nodes to find one or more relay-peers and randomly select one with acceptable forward latency. An efficient relay selection algorithm is helpful to reduce latencies of the connection establishment, transport and relay-handoff process in ubiquitous and pervasive networks (Ren et al., 2006).

In the ubiquitous and pervasive networks, ICE solution is similar to the well known NAT types that support UDP. In other words, ICE supports UDP transport only. For ICE to work properly, both caller and callee peers must support ICE client, and if a relay server is behind NATs, the relay server must support ICE client too. In a P2P-based ubiquitous and pervasive environment, any server facility should not be fixed, and there should be one more super-nodes acting as the role of STUN server. For this reason, it should be deployed only in a homogeneous and controlled environment. Furthermore, it is likely that the connection establishment will be

delayed because of the process involved in data path discovery by the caller and callee.

3.4. Summary on the limitation of current NAT traversal solutions

We have summarized the current NAT traversal solutions, which use the following strategies:

(1) **Avoiding address translation**: NATs assign a public IP address to endpoints or setup a static mapping temporarily. The method such as RSIP and UPnP are seemed useless to decrease the IP address consumption.
(2) **Forwarding**: Forwarding through a third party relay server (node) is effective but inefficient, the method such as TURN will increase the cost of deployment.
(3) **Try and error**: The endpoints attempt to punch a hole in the NATs helped by an external server, and then the connection from outside can enter via the "hole" like STUN, ICE, etc. However, both of them have one critical defect which is no support for TCP. ICE has the feedback if TCP hole-punching is failed, but STUN does not have any feedback mechanism (Cooper and Matthews, 2006). ICE-TCP and STUNT (Simple traversal of UDP through NATs and TCP) are under development; however, most critics thought that the future of both is unclear. Why STUNT and ICE-TCP cannot traverse traditional NATs effectively? It is not due to the imperfectness of STUNT and ICE-TCP, it is the NATs itself.
(4) **Payload translation**: Some protocols of applications are defined to initiate a reverse connection by specific IP address field in the payload of the packet. Setup an application level gateway (ALG) to translate the IP address will help applications traverse NATs. This method usually supports only specific application such as FTP_ALG and SIP_ALG (Srisuresh et al., 1999).
(5) **Redesign NATs themselves**: To build-up a newer NAT function embedded in network devices seems very costly. However, perhaps it is worth to make a great effort to overcome the NAT problem all at once.

The resources in the ubiquitous and pervasive devices are limited. Attempt of building up a total solution for traversing all existing NATs usually needs large programming effort, complex compilation, and high computing power. These costs are hard to be justified.

4. Next generation NAT framework

All aforementioned problems are pointing to the traditional NATs themselves. Since NAT function is still not defined precisely by standard origination, e.g. IETF, in this article we propose a design of next generation NAT (NATng). We modify the idea of RFC 2694 – "DNS extensions to Network Address Translators (DNS_ALG)" (Srisuresh et al., 1999); this document identifies the need for DNS extensions to NATs and outlines how a DNS Application Level Gateway (DNS_ALG) can meet the need: DNS_ALG modifies payload transparently to alter address mapping of hosts as DNS (Mockapetris, 1987a) packets travel from one address domain to another. This document suggests that DNS_ALG working with a NAT is simplified because all host addresses in private network are bound to a single external address. The DNS name lookup for private hosts (from external hosts) do not mandate fresh private-external address binding, as all private hosts are bound to a single pre-defined external address. However, reverse name lookups for the NAT external address will not map to any of the private hosts and will simply map to the NATs. RFC 2694 also illustrates the operation of DNS_ALG with specific examples including a prototype for bidirectional NATs (BNAT).

RFC 2694 did not solve any NAT problem, but it provides a deep thinking via the concept: if there is a control protocol between DNS_ALG and NATs (Bless, 2008) we can make all private hosts own its unique fully qualified domain name (FQDN) in the Internet. When an external host make a DNS query to DNS_ALG, DNS_ALG will send a (hole-punching) control message to NAT immediately. The NAT now will bind a NAT record to let the external host mapped to the private host which the FQDN directs to. Afterward when the request packet sent by external host reaches the NAT, it will be translated and forwarded to the private host smoothly.

The essence of NATng framework is based on bidirectional NATs, which consists of two major components: (1) a bidirectional NATs, and (2) a DNS_ALG for providing private address name resolutions and hole-punching control function. Both above components coordinate and provide bidirectional access capacity between intranet and extranet (Internet), the device is capable of supporting private IP address sharing a single public IP address to access the whole Internet (through internal to external) based on traditional NAT mechanism; It is also capable of supporting public IP address passing and sharing single private IP address to access the whole intranet via fully qualified domain name (FQDN) addressing mechanism (through external to internal). In other words, the NATng can be allowed to share with as well as to have its own unique FQDN over the Internet for those hosts with private IP address. Theoretically, the number of FQDN and its sub-domain are huge. This mapping method relieves the limitation of IPv4 address space from 2^{32} to 2^{48} and more. It is very helpful to solve the IP address exhaustion problem on the Internet.

The aforementioned control protocol between DNS_ALG and BNATs is called BNATCP (Border Network Address Translator Control Protocol). We can obtain it by extending ICMP (Postel, 1981; Deering, 1991) or defining a new one such as Next Steps in Signaling (NSIS) (Pashalidis and Tschofenig, 2007; Sanda et al., 2008) Layer Protocol (NSLP) (Stiemerling et al., 2008), Host Identity Protocol (HIP) (Stiemerling, 2008; Komu et al., 2009), and Path-coupled Signaling Protocol (Martin et al., 2005), which describe an end-application trigger or triggered manner for NAT traversal. The detail connection process is illustrated in Fig. 2. We adopt a method named "DNS ascent query" which is based on standard DNS query (Mockapetris, 1987b; Gulbrandsen et al., 2000) to trigger the hole-punching control protocol: if client A sends a DNS query for the lookup of client B (behind NATs) and finds that the DNS replies a private IP address (e.g. 192.168.*x*.*x*), client A will remove the current domain level underneath and sends a DNS query again until the DNS replies an illegal IP address. Afterwards client A gets at least two IP addresses, one for external IP, and the other for internal IP of the client B. Then client A has enough information to initiate a BNATCP packet to NATng which locates in front of client B. The detail fields of the control protocol are omitted here.

The deployment cost of above DNS_ALG is truly inexpensive. The only cost for using this solution is (1) a new NAT with that control protocol, called next generation NAT (NATng); (2) all external hosts trying to access the private hosts behind NATs now need to support the control protocol to manipulate NATng, or their DNS server addresses can point to the DNS_ALG with the control protocol.

The alternative method to use the BNATCP is shown in Fig. 3. In this sample we modify "call by FQDN" to "negotiation by middle-box". As the same results, clients in both sides may get the information from middle-box and send BNATP packet to the BNAT of the opposite side, or middle-box may pre-serve the job for both of them.

Let us imagine what the ubiquitous and pervasive computing environment looks like in a smart home. We have intelligent refrigerator, intelligent washing machine and other Internet appliances. All of those devices may be assigned a private IP address because public IP addresses may not be sufficient for our needs. In traditional NAT environment, all those devices must include a NAT

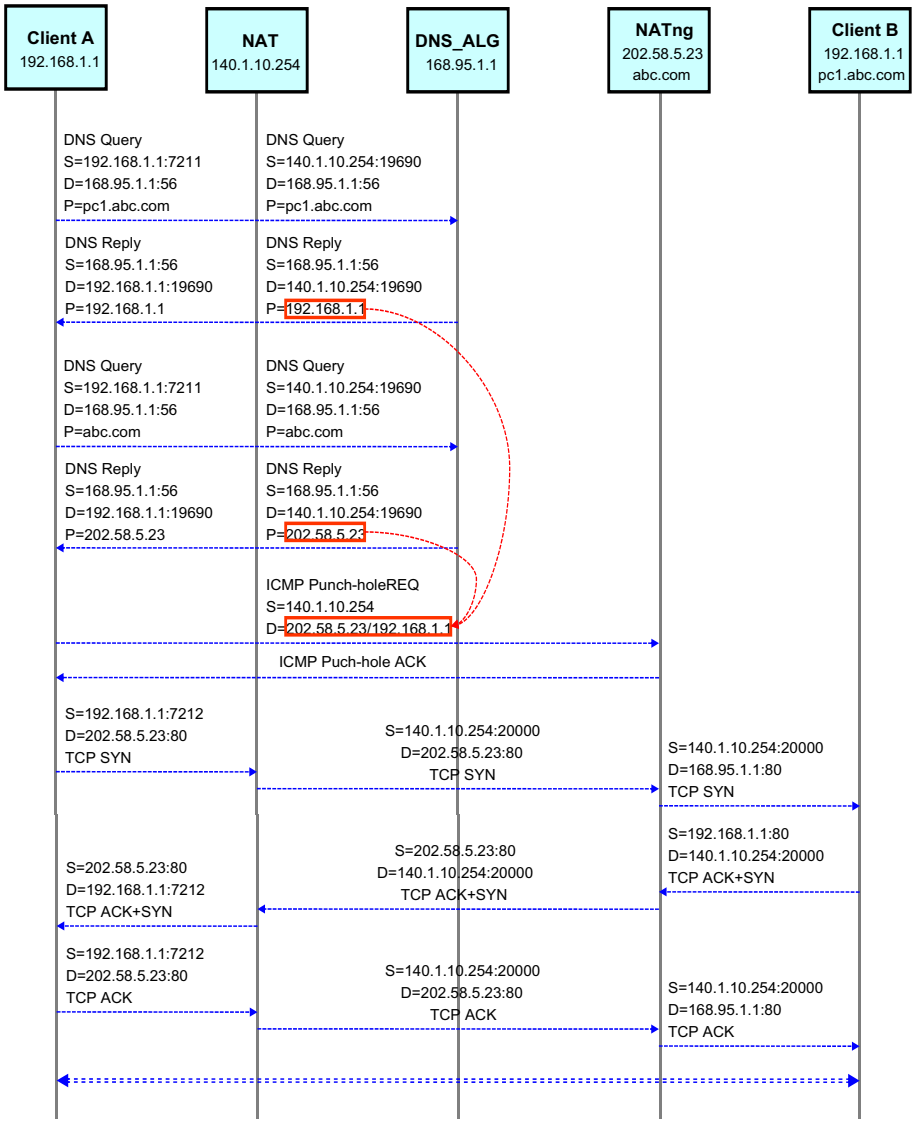

Fig. 2. Packet flow of the NATng operating process between DNS_ALG and NATs – an example of web service.

traversal software agents and 24 hours standby for serving a request. When a request arrives, a complex NAT traversal procedure must be performed first, this may cause the user to feel the lagged response, therefore it would be unable to meet the good "Quality of Experience (QoE)" of smart home.

Look back on our NATng mechanism in a smart home: all those devices could be sleeping for saving power, any requester should take control the NATng first by the BNATCP packet and send the request packet in, then the device will be awaken by the arrived request. The end device neither concerns the NAT problem nor embeds any NAT traversal software agent. NATng will be more relevant with the requirements of ubiquitous and pervasive applications nowadays.

Typically, NATs exist at the border of a stub domain, and hide private addresses from public addresses. With the NATng, the border consisting of NATs will be no longer required. The Internet is usually expressed as a cloud, and this cloud can be enlarged by our new NATng framework.

5. Performance evaluation

Logically, the limitation of IPv4 address space can be relieved if the aforementioned NATng is available. At present, there are no more than 2^{32} connections (limited by association of source/destination port) and 2^{24}-plus expansion address space (actually, RFC1918 space is 17,891,328 addresses, which exist in the form of FQDN) behind each public IP address. Under the rigorous views, the number of connections may decrease to 2^{16}-minus (IP protocol suite identify the source port only), and the address space may be expanded from 2^{24} to 2^{48} in existing IPv4 Internet. Under rough views, both connections and address space will be expanded from 2^{56}, 2^{64} to even infinity (we can cascade multiple BNATs) and all of these addresses can be accessed by FQDN.

To use NATng we only need to pay a little price. For establishing a connection, all applications must adopt call-by-FQDN instead of call-by-IP. The clients don't need to concern what's NAT type and who is behind. Since the standardized hole-punching mechanism has been constructed in all NATng, all NATs have only one unique type henceforth. In the migration period, NATng is also indicated by ICMP or other protocols. In other words, the controller would not be confused with traditional NATs. Indeed, there are still minority of obsolete protocols that contain the IP address in their payload and cause them not suitable in NATng environment. However, we thought that case was rare exception and can be ignored.

Using NATng to establish a connection that traverses the BNATs will result in a little increase in the latency for applications comparing with NAT-free environment. That's because the extra trigger

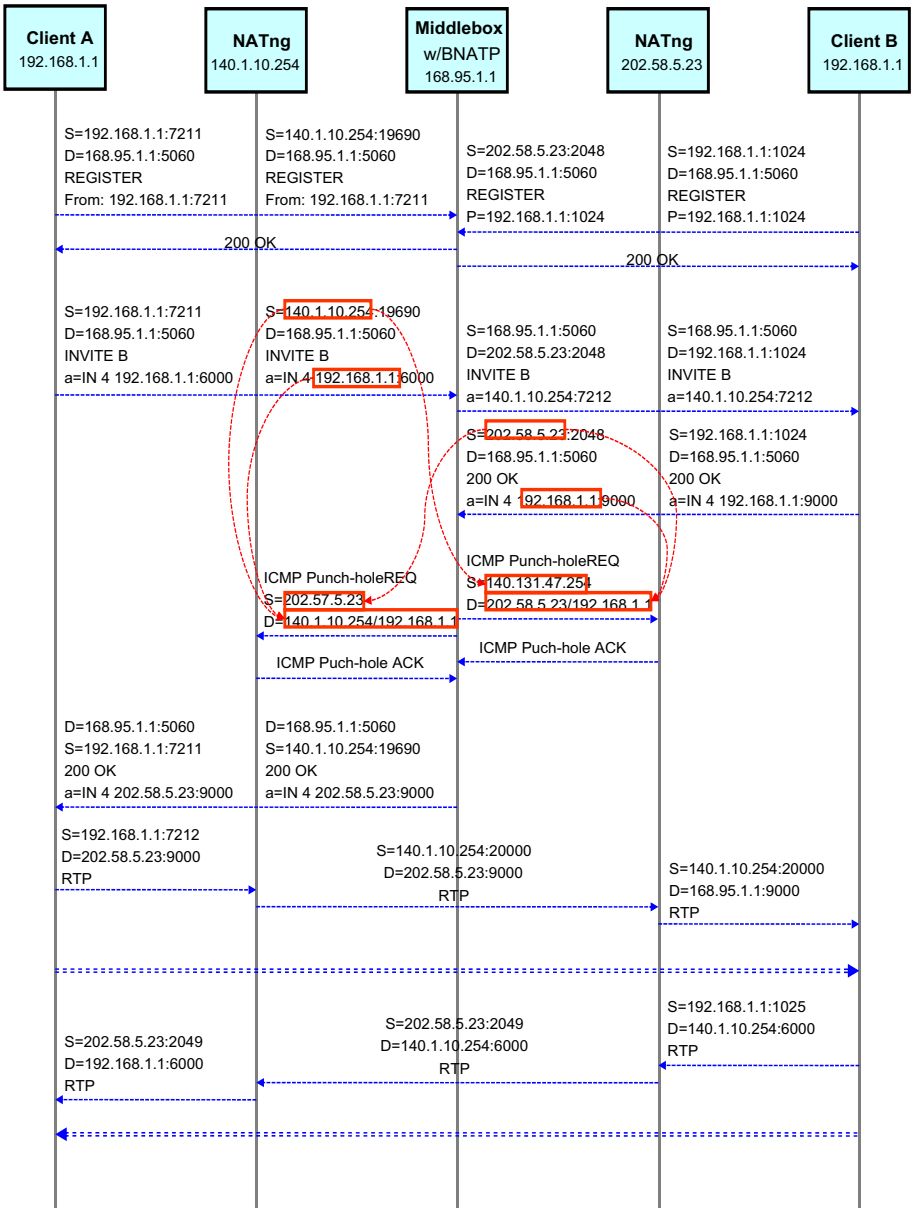

Fig. 3. Packet flow of the NATng hole-punching by middlebox during a session establishment – an example of SIP call.

message will be postponed for only one RTT between DNS_ALG and BNATs. However, NATng still has the advantage of short establishment latency comparing with the STUN/ICE methods. The use of STUN/ICE to discover address-bindings and hole-punching will result in 4–8 messages or more (Rosenberg, 2003), and we cannot be sure whether the traversal is done. For example, a Voice over IP (VoIP) application will suffer an increase of call setup delays, which are at least 4–8 RTTs, to the STUN server (Baldi, 2008).

6. Conclusions

We may face a trade off in using NAT. While it brings the advantage of saving IPv4 addresses, it also destroys the Internet end-to-end transparency. It increases the complexity of applications design, whereas also hinders the services provided in the Internet (Carpenter, 2000).

The problems of NAT are stem from the user's applications or the communication protocols, in which their assumption of network structure is no longer existing (Chown, 2006). The NAT problem has been kept unsolved and not been standardized since the 1980s. We assume that the reason is because the usage of NAT in American and European countries is much less than that in Asia and Pacific areas. Therefore the problem of NAT is not mission critical. In fact, most of developments of network/transport layer and even application layer protocol are available later than the NAT problem, but they overlook the existence of NAT. We think that it is not proper ignoring the NAT problem (Hain, 2000).

Recently, many researches attempt to model and classify the behaviour of NAT in order to construct an effective NAT traversal mechanism. Their efforts have achieved certain success for the UDP, but it still did not accomplish significant achievement regarding the TCP. For the next generation protocols such as SCTP and DCCP, studies concerning about how they interwork with NAT are quite few. These new transport-layer protocols neither consider the NAT problems, nor standardize the traversing mechanism of NAT, not mention to integrate the procedure of connection establishment.

We may imagine that this kind of work is a real challenge (or perhaps there is no solution existed) (Srisuresh and Ford, 2009).

If so, we must furthermore redefine the newer NATng equipment's functions and redesign the NAT-related products. All these new generations of NATs can be controlled by upper layer applications or can be cooperated with. A standard NAT which processes traversing function to perform hole-punching during the connection establishment may be the best solution. For most currently installed traditional NAT devices all those functions may also be obtained through the function of firmware upgrade.

There are too many Internet users and service providers suffering from the NAT problems for a long time. In the future, NAT problem may affect the advance of ubiquitous and pervasive services and applications continuously. Attempting to traverse all traditional NATs has been proved infeasible, at least in TCP. Can we do something to improve the disorderliness? It may be the time to establish a standard NATng mechanism; it is better late than never.

Acknowledgement

We would like to thank the ROC National Science Council for the grant numbered NSC 96-2752-E-009-006-PAE.

References

Aboba, B., Dixon, W., 2004. IPsec-Network Address Translation (NAT) Compatibility Requirements. IETF RFC 3715.
Audet, Ed. F., Jennings, C., 2007. Network Address Translation (NAT) Behavioral Requirements for Unicast UDP. IETF RFC 4787.
Baldi, M., 2008. Providing End-to-End Connectivity to SIP User Agents Behind NATs. IEEE International Conference on Communications (ICC'08), Beijing, China.
Bless, R., 2008. An Explicit Signaling Target Message Routing Method (EST-MRM) for the General Internet Signaling Transport (GIST) Protocol. IETF Internet-Draft. <http://tools.ietf.org/html/draft-bless-nsis-est-mrm-01>.
Borella, M. et al., 2001a. Realm Specific IP: Framework. IETF RFC 3102.
Borella, M. et al., 2001b. Realm Specific IP: Protocol Specification. IETF RFC 3103.
Boulton, Ed. C. et al., 2008. Best Current Practices for NAT Traversal for SIP. IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-sipping-nat-scenarios-09>.
Carpenter, B., 2000. Internet Transparency. IETF RFC 2775.
Chown, T., 2006. The Cost of Application Development with NATs. IETF Internet-Draft. <http://tools.ietf.org/html/draft-chown-cost-of-nat-00>.
Cooper, E., Matthews, P., 2006. Eliminating Duplicate Connectivity Checks in ICE. IETF Internet-Draft. <http://tools.ietf.org/html/draft-matthews-mmusic-ice-eliminating-duplicates-00>.
Deering, S., 1991. ICMP Router Discovery Messages. IETF RFC 1256.
Egevang, K., Francis, P., 1994. The IP Network Address Translator (NAT). IETF RFC 1631.
Guha, S. et al., 2008. NAT Behavioral Requirements for TCP. IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-behave-tcp-08>.
Gulbrandsen, A. et al., 2000. A DNS RR for specifying the location of services (DNS SRV), IETF RFC 2782.
Hain, T., 2000. Architectural Implications of NAT. IETF RFC 2993.
Holdrege, M., Srisuresh, P., 2001. Protocol Complications with the IP Network Address Translator. IETF RFC3027.
Jennings, C., 2007. NAT Classification Test Results. IETF Internet-Draft. <http://tools.ietf.org/html/draft-jennings-behave-test-results-04>.
Johnston, A., Sinnreich, H., 2006. SIP, P2P, and Internet Communications. IETF Internet-Draft. <draft-johnston-sipping-p2p-ipcom-02>.
Komu, M., et al., 2009. Basic HIP Extensions for Traversal of Network Address Translators. IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-hip-nat-traversal-06>.
Martin, M. et al., 2005. Path-coupled signaling for NAT/firewall traversal. In: Workshop on High Performance Switching and Routing, 2005 (HPSR 2005). Hong Kong, China. 231– 235.
Mockapetris, P., 1987a. Domain Names – Concepts and Facilities. IETF RFC 1034.
Mockapetris, P., 1987b. Domain Names – Implementation and Specification. IETF RFC 1035.
Pashalidis, A., Tschofenig, H., 2007. GIST NAT Traversal. IETF Internet-Draft. <http://tools.ietf.org/html/draft-pashalidis-nsis-gimps-nattraversal-05>.
Postel, J., 1981. Internet Control Message Protocol. IETF RFC 792.
Quittek, J. et al., 2006. Problem Statement for SIP-signalled Peer-to-Peer Communication across Middleboxes. IETF Internet-Draft. draft-<http://tools.ietf.org/html/draft-quittek-p2p-sip-middlebox-00>.
Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G.J., Lear, E., 1996. Address Allocation for Private Internets. IETF RFC 1918.
Ren, S., Guo, L., Zhang, X., 2006. ASAP: an AS-Aware Peer-relay protocol for high quality VoIP. In: Proceedings of the 26th International Conference on Distributed Computing Systems (ICDCS'06), Lisbon, Portugal.
Rosenberg, J., 2003. Interactive connectivity establishment (ICE): a methodology for network address translator (NAT) traversal for the session initiation protocol (SIP). IETF Internet-Draft. <http://tools.ietf.org/html/draft-rosenberg-sipping-ice-01>.
Rosenberg, J., et al., 2003. STUN- Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). IETF RFC 3489.
Rosenberg, J., et al., 2008. Session Traversal Utilities for NAT (STUN). IETF RFC 5389.
Rosenberg, J., Mahy, R., Matthews, P., 2009. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN). IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-behave-turn-14>.
Saha, D., Mukherjee, A., 2003. Pervasive computing: A paradigm for the 21st century. IEEE Computer 36 (3), 25–31.
Sanda, T. et al., 2008. Applicability Statement of NSIS Protocols in Mobile Environments. IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-nsis-applicability-mobility-signaling-10>.
Satyanarayanan, M., 2001. Pervasive computing: Vision and Challenges. IEEE Personal Communications 8 (4), 10–17.
Senie, D., 2002. Network Address Translator (NAT) – Friendly Application Design Guidelines. IETF RFC 3235.
Srisuresh, P., et al., 2002. Middlebox communication architecture and framework. IETF RFC 3303.
Srisuresh, P., Ford, B., Kegel, D., 2008. State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs). IETF RFC 5128.
Srisuresh, P., Egevang, K., 2001, Traditional IP Network Address Translator (Traditional NAT). IETF RFC3022.
Srisuresh, P., Holdrege, M., 1999. IP Network Address Translator (NAT) Terminology and Considerations. IETF RFC 2663.
Srisuresh, P., et al., 1999. DNS Extensions to Network Address Translators (DNS_ALG). IETF RFC 2694.
Srisuresh, P., Ford, B., 2009. Unintended Consequence of two NAT deployments with Overlapping Address Space. IETF Internet-Draft. <http://tools.ietf.org/html/draft-ford-behave-top-06>.
Stiemerling, M., 2008. NAT and Firewall Traversal Issues of Host Identity Protocol (HIP) Communication. IETF RFC 5207.
Stiemerling, M. et al., 2008. NAT/Firewall NSIS Signaling Layer Protocol (NSLP). IETF Internet-Draft. <http://tools.ietf.org/html/draft-ietf-nsis-nslp-natfw-20>.
Swale, R.P. et al., 2002. Middlebox Communications (midcom) Protocol Requiremen. IETF RFC 3304.
Weiser, M., 1991. The computer for the 21st century. Scientific American 265 (3), 66–75.

Yaw-Chung Chen received his B.S. degree from National Chiao Tung University, Hsinchu, Taiwan, his M.S. degree from Texas A& M University, Kingsville, Texas, USA, and his Ph.D. degree from Northwestern University, Evanston, Illinois, USA. In 1986 he joined AT & T Bell Laboratories, where he worked on various exploratory projects. He joined National Chiao Tung University, Hsinchu, Taiwan as an associate professor in 1990. He is currently a professor of Computer Science and Information Engineering and Dean of College of Computer Science, Asia University, Taichung, Taiwan. He is also acting as director of Information Industry Institute/National Chiao Tung University Joint Research Center. His research interests include live media streaming, mobility management, P2P systems and green computing. He is a senior member of IEEE and a member of ACM.

Wen-Kang Jia received his B.S, degree in industrial management from the Chung-Hua University, and the EMBA degree in information management from Ming-Chuan University. He is also a Ph.D. candidate with the Department of Computer Science, National Chiao-Tung University, Hsinchu, Taiwan. He has fifteen years of networking experience in a variety of fields including network integration, IP protocol design, VoIP technology, NAT traversal, quality of service, and teletraffic engineering. He was previously a senior professional manager of the IT, Manufacture, Telecom, and Network Industry.