NAT Traversal Techniques: A Survey

Farida Chowdhury
Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet, Bangladesh

ABSTRACT

Network Address Translation (NAT) is one of the most widely-used Ad-hoc techniques in the world. Its soul purpose has been the effective utilisation of IPv4 public addresses by enabling the sharing of a single (or few) IPv4 address(es) by a large number of nodes within a private network. Since its inception, it has achieved a wide-scale adoption worldwide. Unfortunately, it imposes a great obstacle with respect to Peer-to-Peer (P2P) applications. To address this issue, different NAT traversal techniques have been proposed. This paper presents a survey of different NAT traversal techniques from classical solutions to non-standardized solutions. For each technique, their mechanisms, strengths and limitations are explored. Finally, this paper presents the findings in tabular formats so as to provide a side-by-side comparison of different NAT traversal techniques.

General Terms

Networking, Peer-to-Peer (P2P) Networking

Keywords

Network Address Translation (NAT), P2P, STUN, TURN

1. INTRODUCTION

Addressing is a core functionality of the IP (Internet Protocol) in the TCP/IP protocol stack which allows to uniquely identify a device on the Internet. In the earlier version of the IP (known as Internet Protocol Version 4 or *IPv4*, in short), 32 bits of spaces have been allocated for the addressing scheme. This enables it uniquely address around 2^{32} devices. During the earlier design period this addresses was thought to be enough, however, with the exponential expansion of the Internet, an increasing amount of devices have been connected to the Internet resulting in the exhaustion of unique IPv4 addresses [1].

To remedy the situation, a new version of IP (known as Internet Protocol Version 6 or *IPv6*) with a 128 bit address scheme has been introduced supporting 2^{128} bits of unique addresses. It has been hoped that everyone would start adopting the IPv6 addressing because of its expanded addressing scheme and other advantages. The deployment of the Internet Protocol Version 6 (IPv6) has been relatively slow in the Internet, hence the majority of the networks are still IPv4-based, thus, creating a bottleneck. As an intermediary solution, the concept of NAT (Network Address Translation) has been introduced which facilitates an effective utilisation of IPv4 public addresses by enabling the sharing of a single (or few) IPv4 address(es) by a large number of nodes within a private network. Furthermore, the utilisation of NAT has helped to increase the security of private networks. Because of these reasons, NAT has been widely adopted all over the world.

Unfortunately, NAT has also introduced some new obstacles, particularly, with respect to establish and maintain continuous connection between two devices, residing under NATed private networks, as required by Peer-to-Peer (P2P) applications. To tackle this obstacle, a number of NAT traversal techniques have been introduced which have been adopted in a number of application domains such as P2P messaging, P2P file sharing, P2P streaming and so on. These techniques vary in their internal mechanisms and not all techniques are suitable for all application domains. To judge the suitability for any particular application domain, it is important to fully comprehend the internal mechanisms of the proposed NAT traversal techniques. The principal motivation of this article is to aid towards this aim.

This article provides a concise survey of a large number of NAT traversal techniques from classical solutions to non-standardized solutions. For each technique, their mechanisms are explored, their strengths are analysed and the corresponding limitations are outlined. Finally, this paper presents the findings in tabular formats so as to provide a visual side-by-side comparison of different NAT traversal techniques.

Structure: The article is structured as follows. Section 2 briefly explains NAT along with its different terminologies. Section 3 presents the review of different NAT traversal techniques. Section 4 concludes the article.

2. OVERVIEW OF NAT AND NAT TERMINOLOGY

NATs are used to provide a mapping of a single public IP address onto several end systems on a private network, thereby allowing many computers in the private network to access the Internet using the single public IP address [2]. The basic idea is to let a NAT based router replace the IP header of the packets and maintain a mapping table that contains the address information in outgoing and incoming messages. Figure 1 shows the general NAT approach.

2.1 NAT Terminology

Depending on the address mapping schemes, there are four types of NAT: Full Cone NAT, Restricted Cone NAT, Port Restricted Cone NAT and Symmetric NAT [3].

Full Cone NAT: In a full cone NAT, all requests from an internal IP address and port are mapped in the same public external IP

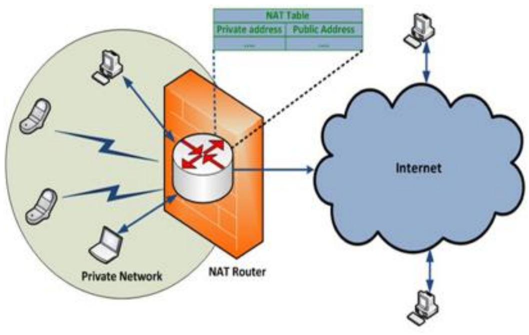

Fig. 1. Network Address Translation (NAT).

address and port. Additionally, other external hosts can send a packet to the internal host using the mapped external address. Figure 2 shows an example of full cone NAT.

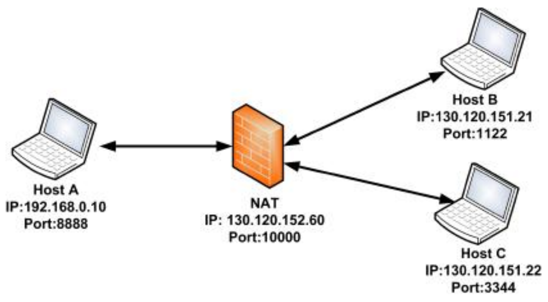

Fig. 2. Full Cone NAT.

Restricted Cone NAT: A restricted cone NAT is like a full cone NAT; the only difference is that an external host with the IP address X can send a packet to an internal host only if the internal host previously sent a packet to the host with the IP address X. Figure 3 shows the behaviour of the restricted cone NAT.

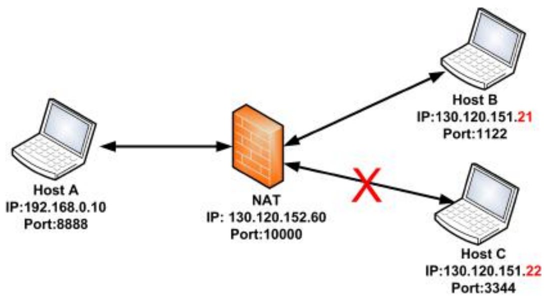

Fig. 3. Restricted NAT.

Port Restricted Cone NAT: A port restricted cone NAT is similar to the restricted cone NAT. Additionally, it requires the external host to use the same port number that the internal host previously used to contact it. Figure 4 shows an example of a port restricted cone NAT.

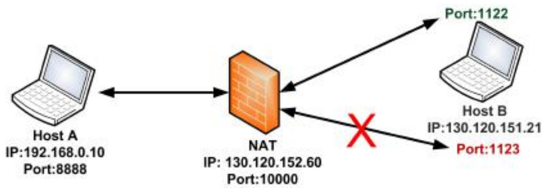

Fig. 4. Port-Restricted NAT.

Symmetric NAT: A symmetric NAT creates a different IP address and port number mapping according to a session IP and an arbitrarily chosen externally used port number. Figure 5 illustrates the behaviour of the symmetric NAT.

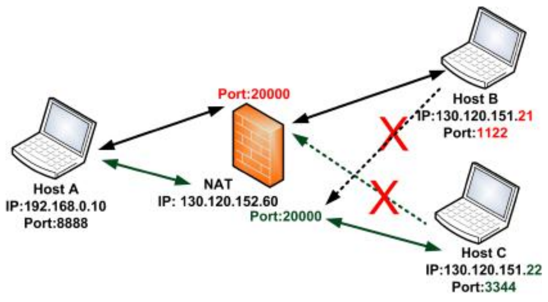

Fig. 5. Symmetric NAT.

2.2 Hairpining

Hairpin or NAT Loopback translation is used when there is a situation which requires two hosts behind the same NATed network to exchange messages via NAT traversal. This can be achieved when it is possible to relay packets between such NATed nodes.

2.3 Challenges with NAT

NAT is suitable for any typical Client-Server communication when the client is on a private network and the server is in the global address realm. The problem arises when it is direct communication or P2P communication as P2P is different from the Client-Server network. In a P2P network peers have equal positions without any classification of client and server. They are directly connected to other peers and they both act as client and server simultaneously. In NATed environments, any general NAT architecture makes it difficult for two nodes on different private networks to communicate with each other directly. Therefore, it is important for NAT device makers, protocol designers and P2P application vendors to provide smooth and secure two way direct communication, including unsolicited incoming connection attempts for hosts residing in NATed environments. The way it is done is known as the NAT Traversal technique. There exists an array of NAT Traversal Techniques offering transparent traversal abilities to keep the P2P connection alive.

3. NAT TRAVERSAL SOLUTIONS

Different NAT traversal mechanisms have been designed to provide direct communication between peers behind NATs, however, none of them provides a solution that works well with all types of NATs. Some are based on NAT gateway optimised and plugged techniques such as Universal Plug and Play (UPnP) [4] and Application Lever

Gateway (ALG) [5], whereas some are based on fall-back (making use of the Client-Server model) approaches, in which they depend on a relay server or a rendezvous server on either or both sides of the NAT gateway. Some examples of such techniques are STUN [3] and TCP/UDP hole punching [6].
This paper has categorised the available NAT traversal solutions among eight categories which are given below:

i) NAT Traversal for UDP Traffic
ii) NAT Traversal for TCP Traffic
iii) NAT Traversal for combined UDP and TCP
iv) NAT Traversal for Structured P2P Overlays
v) NAT Traversal with Cooperative NATs
vi) Combination of Techniques
vii) Traditional NAT Traversal Techniques
viii) Others

The next subsections provide a brief review of a number of currently known NAT traversal techniques for implementing P2P applications like online gaming, media streaming and so on to manage direct communication over existing NAT devices according to the categories described above.

3.1 NAT Traversal for UDP Traffic

3.1.1 STUN. STUN (Simple Traversal of UDP through NAT) was first defined in RFC 3489 [3] in 2003, and then revised in RFC 5389 [7] in 2008 and again in RFC 8489 [8] in 2020.
The classification of NAT types defined in RFC 3489 was found to be faulty, as many NATs available in the market did not fit properly. To resolve this issue, STUN was modified in RFC 8489 to add attributes to binding messages and defined as Session Traversal Utilities for NAT (STUN).
STUN is a simple client-server protocol that is a well known and the most widely used VoIP NAT traversal solution for UDP traffic. It allows applications to discover the presence and types of NATs between them and the public IP address.

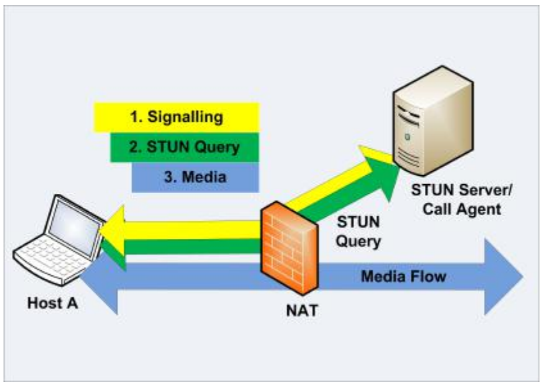

Fig. 6. STUN.

A possible STUN configuration has been shown in Figure 6. The host behind a NAT is called a STUN client, and the server on the external side is called a STUN server. The STUN client sends a Binding Request over UDP to the STUN Server which typically resides in the public address realm. The UDP request packet may traverse several NAT devices to reach the STUN server. The Server discovers the last NAT-modified source address and port and then copies the source address and port into a Binding Response which is then sent back to the STUN client. By comparing the local address and port in the response packet with its own record, the STUN client can discover if it is behind a NAT device.
When the client detects that it is behind a NAT, it does a series of tests to determine its exact type. These tests consist of asking one or two STUN servers to send their responses from different ports and analysing these responses to determine how the NAT has mapped each outgoing request. After detecting the existence and type of the NAT, the client uses the mapping that the NAT allocated for the STUN server to construct its messages.
The main advantage of using STUN is that it does not need any changes to NAT devices. Clients can discover NAT devices automatically. However, it does not support symmetric NATs which are reasonably common. STUN requires client applications to be upgraded to support STUN and an additional STUN server residing in the public domain. The use of STUN can be hindered due to these reasons.

3.1.2 UDP Hole Punching. The Hole Punching technique [6] enables direct P2P communications between hosts or peers, even if the peers are both behind NATs, with the help of a well-known rendezvous server. The rendezvous server allows the peers to discover each others' endpoints (IP address and port) so that they can communicate directly.

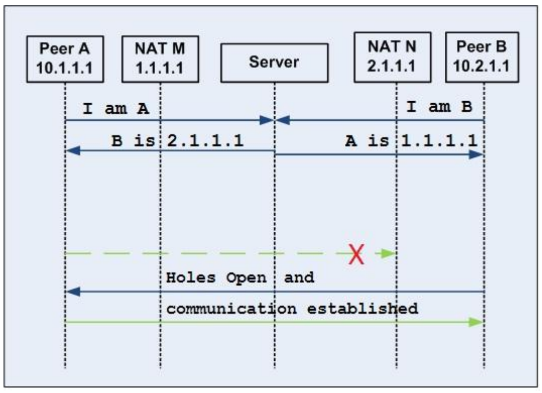

Fig. 7. UDP hole punching.

Figure 7 shows an example of the UDP hole punching process. Suppose peers A and B are both behind different NATs M and N respectively. A and B have established a UDP session to the server and, the NAT M and N will create UDP translation states and assign a temporary external port number for each internal peer. Afterwards, the server will relay the information containing their IP address and port number back to peer A and peer B.
Now peer A sends packets to the public address of the peer B. Since the remote NAT N does not have a translation entry for this connection yet, it drops all messages. Now B sends data packets to the public address of A, to the exact IP address and port where the previous message came from. Although the message of A was dropped, N created a translation entry on its NAT device which now allows the message from B to pass. This enables the UDP connection to be established.

3.1.3 UDP Multi Hole Punching. As UDP hole punching technique can not traverse symmetric NAT, authors in [9] proposed a new UDP hole punching technique to traverse symmetric NATs successfully. It uses a new port prediction method that controls and manipulates port numbers in order to traverse symmetric NAT boxes as well as other kinds of NATs.

3.1.4 Teredo. Teredo [10] provides a NAT traversal service for connecting IPv4 nodes, residing behind NAT devices, to IPv6 devices external to the IPv4 nodes. Teredo uses four types of nodes - Teredo clients, Teredo servers, Teredo relays and IPv6 nodes. Teredo tunnels packets over UDP to a Teredo relay node, acting as a Rendezvous server, and thus facilitating traffic relay between NATed IPv4 nodes and external IPv6 devices. Teredo clients can detect the type of the NAT. If it is not behind a symmetric NAT, the process becomes successful.

3.1.5 PS-STUN. PS-STUN [11] stands for Predicting and Scanning STUN. This technique is based on STUN [3] and also can traverse the symmetric NAT. It has defined two different schemes for port assigning in Symmetric NAT discovery. The first scheme is to assign the mapping ports with continuous and progressive numbers which is known as *Progressive* or *P* type Symmetric NAT, whereas the second one is to assign a new mapping port to a random number which is known as *Random* or *R* type Symmetric NAT.
PS-STUN cannot traverse the *R* symmetric NAT and the predicting algorithm work similarly as STUNT (which will be explained in next section). It also uses a central server to traverse the P and R type symmetric NATs.

3.1.6 Summary. Table 1 presents a brief summary of the NAT traversal techniques for UDP traffic along with their advantages and disadvantages. It is to be noted that in the second column of the Table 1, ‘Cone’ means all three types of NAT: full cone, restricted cone and port restricted cone; as described in Section 2.1.

3.2 NAT Traversal for TCP Traffic

3.2.1 STUNT. STUNT (Simple Traversal of UDP through NATs and TCP too) [12] is a protocol that extends STUN to include TCP. It includes two approaches for traversing NATs. The first approach is known as STUNT#1, which has been illustrated in Figure 8. The figure shows that the two peers A and B are behind two different NATs M and N respectively and the STUNT server is on the public Internet. In this approach, both peers send an initial SYN with a high enough TTL to cross their own NATs, however, the TTL is low enough that the packets are dropped in the network (once the TTL expires). The peers learn the initial TCP sequence number by listening for the outbound SYN over PCAP or a RAW socket. Both peers inform the STUNT server of their respective sequence numbers and the STUNT server replies with a SYN-ACK (spoofed) to both peer with the appropriate sequence numbers. The ACK completes the TCP handshake and therefore goes through the network as usual.
This approach has four significant problems. Firstly, it requires the peer to determine a TTL that is large enough to cross its own NATs but low enough not to reach the other peer’s NAT. The problem is that when the two outermost NATs share a common interface, such a TTL does not exist. The second problem is that the ICMP TTL - exceeded error could be generated in response to the SYN packet and this might be interpreted by the NAT as a fatal error. Thirdly, the TCP sequence number of the initial SYN might be changed when it arrives at the NAT. The fourth problem is that it depends on a third-party to spoof a packet for an arbitrary address.
In the second approach, which is known as STUNT#2, it is also required that the STUNT server, which is omitted in Figure 9, discovers the public IP address and port number of the NAT and the NAT type. Unlike STUNT#1, the STUNT server does not need

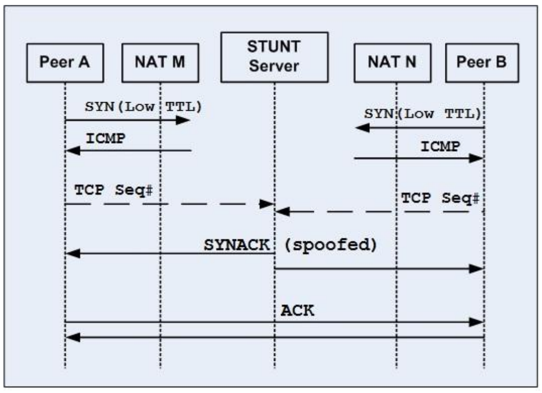

Fig. 8. STUNT#1.

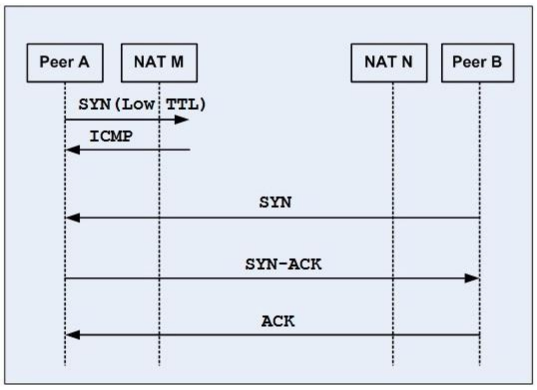

Fig. 9. STUNT#2.

to spoof the SYN-ACK packet and therefore, it does not need to have the root or administrator privilege. Only one peer sends out a low-TTL SYN packet and then terminates the connection attempt and creates a TCP socket (passive) on the same address and port. The other peer then initiates a regular TCP connection, as shown in Figure 9. As with STUNT#1, it is important for the peer to pick an appropriate TTL value and the NAT must not consider the ICMP error to be a fatal error. It also requires that the NAT accepts an inbound SYN following an outbound SYN which is a sequence of packets not normally seen.
The main contributions of the STUNT are to predict the next port number and establish the direct TCP connections.

3.2.2 NATBlaster. NATBlaster [13] is another approach similar to STUNT#1 but avoids the IP spoofing requirement. In this approach, both peers initiate an outbound connection and keep the record of their initial sequence number. The two peers then interchange the sequence numbers and each peer sends a SYN-ACK packet to another. The SYN-ACK packet is sent into the network using a RAW socket. Once the SYN-ACKs are received by both peers, ACKs packets are sent to complete the connection setup. Figure 10 shows the packet flow in NATBlaster.
Like the STUNT#1 approach, NATBlaster also requires the peer to properly select the TTL value, requires the NAT to ignore any ICMP error and it fails if the NAT changes the sequence number of the SYN packet. In addition, it requires the client applications to have access to RAW sockets, which are usually available at root or administrative privilege levels.

Table 1. NAT traversal techniques for UDP.

NAT traversal techniques	NAT types	Advantages	Limitations
STUN [3, 8, 7]	Cone	1) Does not require any changes on NAT devices. Clients can learn NAT devices automatically. 2) Easy and standardised.	1) Short term solution. 2) Requires client application to be upgraded to support STUN and an additional STUN server residing in the public Internet.
UDP Hole Punching [6]	Cone	1) Preserves the transparency of NAT 2) Works even with multiple levels of NATs.	1) By introducing a server, it turns the part of a P2P network into Client-Server model. 2) It adds overhead to the bandwidth and increases communication latency as well.
UDP Multi Hole Punching [9]	Both	1) It solves symmetric NATs. 2) Sends UDP packets with a low TTL value.	1) It uses two servers. 2) It only solves the symmetric NAT with port number increasing sequentially and it does not predict next using port number of the symmetric NAT.
Teredo [10]	Cone	It simplifies the IPv6 deployment process and facilitates IPv6 transition phase.	It does not work for Symmetric NATs.
PS-STUN [11]	Both	It can traverse cone type NAT and the *P* type symmetric NAT successfully.	1 It cannot traverse the *R* symmetric NAT. 2) Uses a central server.

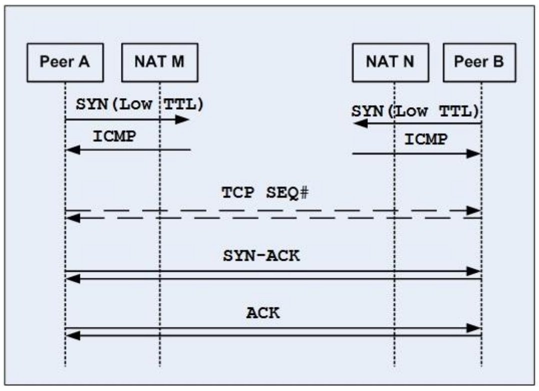

Fig. 10. NATBlaster.

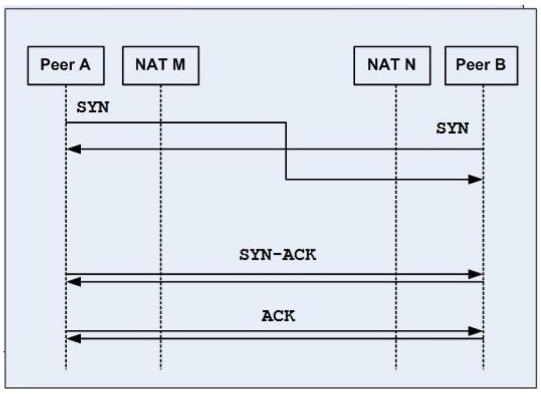

Fig. 11. P2P-NAT.

3.2.3 *Peer-to-Peer NAT.* The mechanism of Peer-to-Peer NAT [6] is similar to STUNT#2. In Peer-to-Peer NAT, both peers send SYN packets and listen for any incoming connections to the same port at the same time. One of the NATs will end up following the simultaneous open sequence, where the other one follows the regular open sequence. Figure 11 shows the packet flow in Peer-to-Peer NAT. Basically this approach is not as popular as STUNT#2 and it additionally requires the peer to retry failed connection attempts until a time-out occurs.

3.2.4 *NatTrav.* NatTrav [14] is also known as sequential TCP hole punching. In NatTrav, there are three types of entities: "recipient" and "initiator" peers as well as a "connection broker". An initiator peer would like to initiate a connection with a recipient peer. On the other hand, a recipient peer is a NATed node that is willing to receive connections from an initiator peer. To facilitate this connection, a broker node is utilised. A recipient node registers, either using TCP or UDP, with a connection broker to receive a unique URI (Universal Resource Indicator). The broker also facilitates NAT traversal for any NATed recipient. An initiator must provide a network address and URI of a recipient to the broker so that it can establish a connection between these two nodes.

3.2.5 *Summary.* Table 2 presents a brief summary of all the NAT traversal techniques under TCP along with their advantages and disadvantages. In the second column of the table, 'Cone' means all three types of NAT: full cone, restricted cone and port restricted cone; as described in Section 2.1.

3.3 NAT Traversal for combined UDP and TCP

3.3.1 *TURN.* In some scenarios, a direct connection between nodes behind different NATs is impossible, particularly if nodes are behind a symmetric NAT. In these cases, it is necessary to relay the communication via an external intermediate node. Traversal Using Relays around NAT (TURN) [15] is a protocol that provides a relaying service via a TURN server. As this approach makes heavy use of relay techniques, it works in almost every imaginable situation. However, this service is expensive to maintain as it is required to maintain a robust and high-capacity TURN server. Consequently, in most situations, TURN is used as a fall-back mechanism for other methods, such as hole punching, to solve the NAT connectivity problem. This approach works both for TCP or UDP. The principle of TURN is to provide a client with a public Internet address, which are usually hidden by NAT devices. This allows the clients to request a public address and port from the TURN server. The TURN server communicates to the TURN client with some requests/responses besides relaying and the server is transparent to external peers. Even so, one client behind the NAT and the TURN server does not receive the messages from the third (other) peers. Unlike STUN, TURN does not allow direct connectivity between two hosts behind their NATs. Although TURN can traverse each

Table 2. NAT traversal techniques for TCP.

NAT traversal techniques	NAT types	Advantages	Limitations
STUNT#1 [12]	All types	Direct TCP connection.	1) Need supervisor privileges. 2) Determining a proper TTL value. 3) Requires IP address spoofing.
STUNT#2 [12]	All types	1) Direct TCP connection. 2) No need for administrator privilege.	Determining a proper TTL value.
NATBlaster [13]	All types	Does not require IP Spoofing.	1) Need supervisor privileges. 2) Determining a proper TTL value.
P2P-NAT [6]	All types	Easy.	1) It is not as popular as STUNT#2. 2) Packet flood may occur.
NatTrav [14]	Cone	1) One single socket is needed to bound to a given local port. 2) The OS does not need to handle simultaneous TCP connection setup. 3) Provides NAT traversal with minimal cost. 4) No need for the users to reconfigure their NATs.	Does not support Symmetric NATs.

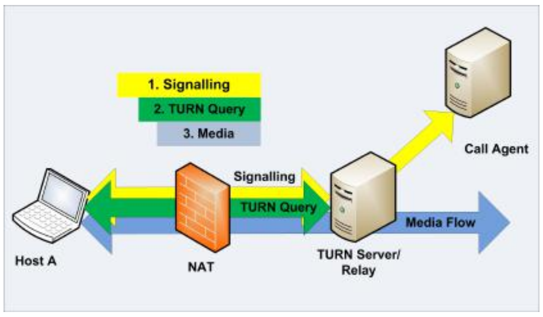

Fig. 12. TURN.

type of NAT including symmetric NAT, the TURN protocol will burden the public TURN server with heavy loads and might cause delays. Due to its associated high cost, TURN will be the last resort to use in practice. The Figure 12 shows the TURN architecture.

3.3.2 TCP Hole Punching . Hole punching is well understood for UDP, but it can also work for TCP communication [6]. To establish a direct P2P TCP connection between peers under NAT is a little more complex than for UDP. It works in a similar way to UDP hole punching, only with the added complexity of establishing the TCP handshake between the peers.
Suppose two peers A and B are both behind different NATs as shown in Figure 13. Similar to UDP, both peers establish an active TCP connection to the same rendezvous server S. The peers also register their private and public addresses on the server. Each peer's first SYN packet to the other peer creates a 'hole' in its respective NAT. In the situation when A's first SYN packet to B reaches at B's NAT before B's first SYN packet to A reaches at B's own NAT, B's NAT tags A's SYN packet as unsolicited and therefore, drops it. However, B's SYN packet can get through A's NAT successfully because B's public address is recognised by A's NAT as part of the outgoing session to B that A had already initiated.

3.3.3 Summary. Table 3 presents a brief summary of all the NAT traversal techniques for the combined UDP and TCP along with their advantages and disadvantages. In the second column of the table, 'Cone' means all three types of NAT: full cone, restricted cone and port restricted cone; as described in Section 2.1.

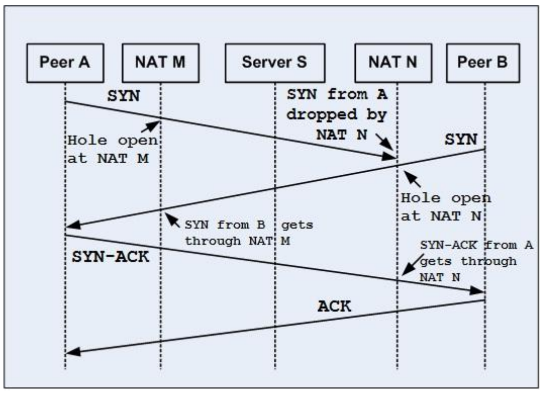
Peer A
NAT M
Server S
NAT N
Peer B
SYN
SYN from A dropped by NAT N
Hole open at NAT M
SYN
Hole open at NAT N
SYN from B gets through NAT M
SYN-ACK
SYN-ACK from A gets through NAT N
ACK

Fig. 13. TCP hole punching.

3.4 NAT Traversal for Structured P2P Overlays

There are a number of attempts have been made on how to traverse NAT in P2P overlays in general, however, a very few number of attempts made to apply NAT Traversal techniques in structured P2P overlays. This section will analyse these techniques.

3.4.1 SMBR. SMBR (Selective-Message Buddy Relaying) [16] uses a server-less distributed NAT traversal mechanism for DHT based P2P overlays. DHT algorithms have two kinds of messages: control messages and data messages. Control messages are small sized packets used to search and publish the resource file, whereas data messages are big sized packets used to transfer the files. SMBR uses different techniques for traversing control messages and data messages. At first, the Private Nodes (PNs), which are behind NATs randomly select buddies, which are Global Nodes (GNs) that have public IP addresses. Once the PNs have found their buddies, they can traverse the control messages via the buddies relay. For traversing the data messages, a direct connection is made with the help of the buddies. If the communication has to be made in between two PNs, the UDP hole punching technique is used. In this case, the traversal time is higher than relaying which is used to traverse the control messages. The authors claim that this is cost efficient as the two nodes send and receive a lot of data after traversal. Again, the load of the buddies is only for sending and receiving few pack-

Table 3. NAT traversal techniques for UDP and TCP both.

NAT traversal techniques	NAT types	Advantages	Limitations
TURN [15]	All types	1) TURN can traverse every type of NAT including symmetric NATs. 2) TURN is used to relay media streaming between peers.	1) TCP data is proxyied by a third party which creates a potential network bottleneck. 2) Requires excessive network resource requirements 3) TURN server must remain available for the whole duration of the allocation. 4) Unlike STUN, TURN does not allow a direct connectivity between NATed hosts.
TCP Hole Punching [6]	Cone	1) Provides a direct TCP connection which is a reliable connection with error detection and packet retransmission. 2) Preserves the transparency of NAT. 3) Works with multiple levels of NATs.	1) By introducing a server, it turns the part of a P2P network into a Client-Server model. 2) It adds overhead to the bandwidth and increases communication latency as well. 3) Additional processes required to establish the TCP handshake as well as to synchronise the packet sequence numbers.

ets and processing these packets, which is much lighter than relay. Therefore the system can achieve a balance between the traversal time and the buddys load. The authors implemented this technique in a prototype system of Kademlia and tested this approach in a smaller LAN environment.

3.4.2 UDP Hole Punching without a dedicated Rendezvous Server. An alternative method of UDP hole punching for structured P2P overlays is proposed in [17] that does not utilise a predefined rendezvous server. Instead, a few randomly chosen public peers are equipped with the required functionalities so that they can emulate themselves as rendezvous servers. The public peers are assumed to have considerably higher lifetime than their private counterparts and are easily reachable from any public or private peer alike. Even so, to safeguard against the sudden unavailability of any chosen emulating rendezvous server, a resilience strategy has maintained.
For each private network, a couple of existing public peers are randomly chosen as its corresponding rendezvous servers. Then, each newly joined private peer in that private network communicates with one of the chosen (emulating) public peers to initiate the hole punching procedure. Two peers are chosen to ensure redundancy as part of the resilience strategy.

3.4.3 Maidsafe. The MaidSafe Platform [18] is a distributed P2P network that allows individuals and companies to create secure, efficient and fast applications to manipulate all kinds of data, share applications, data and communications between individuals and/or groups. It utilises Kademlia based routing and the routing performs DHT hole punching technique to enable direct communication between each pair of nodes in combination with the MaidSafe-RUDP which is an implementation of reliable UDP. Hole punching can be achieved even without the requirement for any servers. But this technique works as long as both communicating nodes are not behind any symmetric NAT. If both nodes are behind symmetric routers, the relay option is used to traverse the NAT. Fairness is an issue in this case where the relay node has to provide more bandwidth to the firewalled node.

3.4.4 Kademlia based NAT Traversal Solution. The authors in [19] proposes a NAT-ed peer organization Model on top of Kademlia [20]. In this model, any two peers can communicate with each other by initiating a message from one open or a NAT-ed peer to another open or a NAT-ed peer. NAT-ed peers can connect to other NAT-ed peers if both of them are NAT traversable. They do not rely on open peers. Each peer has a neighbor list which contains IDs and endpoints of other known open peers or NAT-ed peers. For NAT-ed peers in the neighbor list, a long term session is always maintained by sending keep-alive UDP packets periodically between each other after NAT traversal has been made for direct connection.

3.4.5 Summary. Table 4 presents a brief summary of reviewed NAT traversal techniques for structured P2P overlays along with their advantages and limitations.

3.5 NAT Traversal with Cooperative NATs

A number of NAT traversal methods do not need a server but require modifications on NAT devices. These are explained below.

3.5.1 UPnP. UPnP [4], a new standard, facilitates the NAT traversal for new plugged in devices in the home network. An application in the device can utilise UPnP to configure a NAT box in the home network so that it can receive incoming requests from external networks to a specific port of the device, thus achieving NAT traversal. Unfortunately, the early UPnP implementations were not considered secure and for this, many NAT boxes switch off the UPnP options by default.

3.5.2 ALG. Application Level Gateways (ALGs) [21] are a specific type of translation agents and NAT extension components that are used by specific applications or application layer protocols such as FTP, SIP and file transfer in IM (instant message). Using ALG an application within a host in a private network (presenting an address realm) can connect to its counterpart application in another host in another private network (address realm). To facilitate such a connection, ALG needs to replace the mechanism of existing NAT by reconfiguring static mapping for signal and media stream. For this, ALG needs to reside in the NAT router/device in the network to modify network packets transparently as required. Its requirement of modifying existing NAT behavior is considered as an obstacle for its wide-scale adoption.

3.5.3 MIDCOM. To overcome the limitations of ALGs, MIDCOM [22], (Middlebox Communication), has been proposed which is mostly suitable for providing multimedia services across NAT boundaries. It is also an application specific mechanism suitable for application entities such as VOIP (Voice Over IP) user agents to communicative with the respective server by NAT translation. It is equipped with a firewall and NAT box as well as a MIDCOM agent functioning like a call agent providing SIP and signalling service by which NAT traversal can be carried out.

Table 4. NAT traversal techniques for Structured and Unstructured P2P Overlays.

NAT traversal techniques	Advantages	Limitations
SMBR [16]	Cost efficient.	1) Tested and validated in a small LAN environment. 2) The selection process of buddies is random and does not ensure that it will reach the destination node in a shorter distance.
Maidsafe [18]	UDP hole punching method is used without using any server.	Relaying is used to traverse symmetric NAT.
UDP hole punching without a dedicated server [17]	1) It works in peers behind the same NAT and different NATs efficiently. 2) No need for a dedicated server.	It does not work for peers behind multiple levels of NAT.
Kademlia based NAT traversal [19]	Reliable snd scalable.	It generates a large number of keep alive messages.

3.5.4 NATng. NATng is a next generation NAT framework consisting of a Bi-directional NAT (BNAT), a Domain Name System Application Level Gateway (DNS-ALG) along with a Border Network Address Translator Control Protocol (BNATCP). DNS-ALG is utilised for name resolutions of private addresses and providing hole function capabilities. On the other hand, BNATCP is used to control all BNATs. Leveraging these components, NATng allows all devices under a private intranet to share a single public IP address (as any traditional NAT) and at the same time, enables devices from the public Internet to address and access any device in the private network using a fully qualified domain name.

3.5.5 Summary. Table 7 presents a brief summary of all the NAT traversal techniques under Cooperative NATs along with their advantages and limitations.

3.6 Combination of Techniques

This category provides the available NAT traversal techniques which have implemented using two or more standard NAT traversal solutions.

3.6.1 ICE. ICE (Interactive Connectivity Establishment) has proposed by the Internet Engineering Task Force (IETF) to provide NAT traversal capabilities for session-oriented protocol. It builds NAT traversal intelligence into nodes to perform route discovery, relay lookup, path optimization and media flow verification before a connection is established. It combines the use of STUN [3] and TURN [15] and provides a unified framework around them. ICE hosts exchange accessibility information and negotiate with each other to find one or more communication paths between them.

3.6.2 Combining ICE and SIP. This technique [23] has combined ICE and SIP (Session Initiation Protocol) to traverse NAT. ICE has used for hosts to gather transport addresses from STUN/TURN server and thereby test the connectivity, whereas SIP protocol has used for exchanging these transport addresses between the hosts. This method also works for the new classification of NATs.

3.6.3 Symmetric NAT Traversal with Random Port Assignment. This technique, proposed in [24], integrates four main techniques: STUN [3], STUNT [12], hole punching [6] and P2P-NAT [6]. It can traverse all type of NATs and achieve direct TCP connection between two symmetric NATs with random port assignment.
At first, the method uses the function of STUN that tells if the host behind the NAT and, therefore, it learns the network environment automatically as well as the IP address of the NAT. Then it uses the function of STUNT that distinguishes the type of NATs. For the type of symmetric NAT with random port assignment, it makes a TCP connection to punch a hole on the NAT system as hole punching technique. Finally, it establishes multi-connections to the specific IP address and port number that was assigned randomly by destination NAT.

3.6.4 STUN variant with Superpeers. This technique in [25], implemented a variant of STUN [3] using superpeers. This method first determines the used NAT types for the hosts using STUN and then selects a suitable NAT traversal technique (either Hole Punching [6], Connection Reversal or Relaying) dynamically.
In addition of a STUN server, this method uses new STUN servers which are formed by two peers (called as superpeers) using Full Cone or no NAT at all into the P2P network. In the first phase, each peer needs to determine its own NAT type. UDP based STUN protocol is used to detect how it is connected to the Internet (such as if it is located behind a NAT, the type of the NAT and the hosts external address and so on). In the second phase, an existing NAT traversal technique is used to traverse the NAT. If the peer initiating the communication is behind a Full Cone type and the contacted peer uses one of the more restrictive NAT types, Reversal is used. If both communication peers use Restricted Cone or Port Restricted Cone, the hole punching approach is used. If both communication peers are behind Symmetric, Relaying is used.

3.6.5 Summary. Table 6 presents a brief summary of the reviewed NAT traversal techniques which are the combination of different existing NAT traversal techniques along with their advantages and limitations.

3.7 Traditional NAT Traversal Techniques

3.7.1 Static Mapping. Static Mapping is a manual method to traverse NAT system. To configure this mapping with each private host, an administrator or root privilege is required. This technique is less efficient and not widely used.

3.7.2 Port Forwarding. Port forwarding is a type of NAT traversal technique that assumes direct control over the NAT. Port forwarding entry can be added in the NAT table manually or by dynamically. However, the process of manual entry is an advanced process which can only carried out by experienced users.

3.7.3 Relaying. Relaying is a reliable method of P2P network over NAT. Suppose two private hosts A and B have each initiated

Table 5. NAT Traversal with Cooperative NAT.

NAT traversal techniques	Advantages	Limitations
UPnP	1) It supports automation. 2) It is good for SOHO (Small Office and Home Office) networks.	1) VoIP applications cannot totally rely on UPnP as many user agents and NATs do not support it. 2) It is good for home applications, however, it is rejected by enterprise network administrators due to the security risks it carries.
MIDCOM	It solves the limitations imposed by ALG.	It requires upgradation in existing NATs which is a time-consuming process.
ALG	Transparent for user agents and VoIP servers.	1) ALG processing is complex in managing the address bindings. 2)Less scalable. 3) Slow deployment of new applications.
NATng	Deployment cost is low.	It requires modifications on NAT devices.

UDP or TCP connections to a well-known server S (at global IP address, X: port number, x). If the two hosts reside on separate private networks, their respective NATs prevent each host to initiate a direct connection between them. Without trying for a direct communication, the two hosts use the server S to relay messages between them. Relaying works in all type of NATs and is a useful fall-back technique if maximum robustness is required.

3.7.4 Connection Reversal. Connection Reversal is a straightforward but limited technique. It enables direct communication when both peers are connected to a well-known rendezvous server S and only one of the peers is behind a NAT. If a private host A wants to initiate a connection to a public host B, then a direct communication takes place automatically, as B is not behind a NAT and A's NAT can interpret the connection as an outgoing session. However, if B wants to initiate a connection to the private host A, any direct communication to A is failed by A's NAT. In this case, B relays a connection request to A through the rendezvous server S and asks A to "reverse" a connection back to B.
This method obviously breaks the concept of direct P2P communication but uses a server as an intermediary to set up a direct P2P communication.

3.7.5 Summary. Table 7 presents a brief summary of the reviewed traditional NAT traversal techniques along with their advantages and limitations.

3.8 Others

In this section, a number of non-standardized NAT traversal techniques proposed by different research activities have been reviewed.

3.8.1 CAN. The CAN (Context Aware NAT) [27] implements a software module called User Agent (UA) and installed at the hosts behind NATs to collect their network context information such as host location (public/private domain), NAT type (mapping behavior and filtering behavior), whether hairpin translation is supported or not and whether connection tracking is implemented or not. After that, hosts residing under NAT exchange their network context information to determine the suitable communicating paths.Therefore, it reduces unnecessary checks, shorten check delay and resolves the low DCR (Direct Communication Ratio) problem caused by connection tracking.

3.8.2 ANT. Audio signaling based NAT Traversal or ANT [28] utilizes an innovative technique for NAT traversal in which audio signals are used to establish a connection between two mobile devices with minimal user intervention. For this, ANT does not rely on any intermediate server. Instead, it uses UPnP [4] to obtain NAT configuration information which is then encoded into audio frequencies and then converted and transmitted via the sender's phones. The receiver phone receives the audio signal and converts them back to configuration data which are then used for NAT traversal.

3.8.3 Vsaas. VSaaS (Video Surveillance as a Service) architecture [29] is a WebSocket protocol which can be used for NAT traversal to be used with IP cameras. For this, VSaaS utilises a server and a gateway as an add-on component. The owners of an IP camera needs to install the gateway in the same network as the IP camera. The gateway establishes a bi-directional communication bridge between the IP camera and the VSaaS server by leveraging HTTP-polling and SSH reverse tunneling to traverse the NAT.

3.8.4 LEPaNTU. LEPaNTU (Long polling based Energy efficient Passive NAT Traversal through UDP) [30] is an energy efficient, UDP based NAT traversal scheme. LEPaNTU dynamically determines the lifetime of the respective NAT entry and then periodically sends keep-alive messages to maintain the NAT entry. This results in energy efficiency for receiving on-demand updates of cloud based IoT sensor services.

3.8.5 Summary. Table 8 presents a brief summary of reviewed techniques in this section along with their advantages and limitations.

4. CONCLUSION

In this article, a number of NAT traversal techniques under eight categories have been reviewed. Within each category, the internal mechanisms for a number of major NAT traversal techniques have been briefly presented. In addition, the advantages and limitations have been noted. Finally, a summary of different techniques under each category is presented in a tabular fashion so as to provide a side-by-side comparison of the reviewed works. In summary, the article presents a comprehensive survey of existing NAT traversal techniques. This survey would be beneficial for any researcher who is interested to comprehend these mechanisms and understand their strengths and weaknesses.

5. REFERENCES

[1] APNIC. Ipv4 exhaustion. `https://www.apnic.net/manage-ip/ipv4-exhaustion/`. Accessed: 2020-08-10.

[2] Kjeld Egevang and Paul Francis. The ip network address translator (nat). Technical report, 1994.

[3] Jonathan Rosenberg, Joel Weinberger, Christian Huitema, and Rohan Mahy. Stun-simple traversal of user datagram protocol

(udp) through network address translators (nats). Technical report, RFC 3489, IETF, Mar, 2003.

[4] Mohamed Boucadair, Reinaldo Penno, and Dan Wing. Universal plug and play (upnp) internet gateway device-port control protocol interworking function (igd-pcp iwf). 2013.

[5] Pyda Srisuresh, George Tsirtsis, Praveen Akkiraju, and Andy Heffernan. Dns extensions to network address translators (dns_alg). 1999.

[6] Bryan Ford, Pyda Srisuresh, and Dan Kegel. Peer-to-peer communication across network address translators. In *USENIX Annual Technical Conference, General Track*, pages 179–192, 2005.

[7] Dan Wing, Philip Matthews, Rohan Mahy, and Jonathan Rosenberg. Session traversal utilities for nat (stun). *RFC5389, October*, 2008.

[8] Dan Wing, Philip Matthews, Rohan Mahy, and Jonathan Rosenberg. Session traversal utilities for nat (stun). *RFC8489, February*, 2020.

[9] Yuan Wei, Daisuke Yamada, Suguru Yoshida, and Shigeki Goto. A new method for symmetric nat traversal in udp and tcp. *Network*, 4(8), 2008.

[10] Christian Huitema. Teredo: Tunneling ipv6 over udp through network address translations (nats). Technical report, RFC 4380, February, 2006.

[11] Yong Wang, Zhao Lu, and Junzhong Gu. Research on symmetric nat traversal in p2p applications. In *2006 International Multi-Conference on Computing in the Global Information Technology-(ICCGI'06)*, pages 59–59. IEEE, 2006.

[12] Saikat Guha, Yutaka Takeda, and Paul Francis. Nutss: A sip-based approach to udp and tcp network connectivity. In *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, pages 43–48. ACM, 2004.

[13] Andrew Biggadike, Daniel Ferullo, Geoffrey Wilson, and Adrian Perrig. Natblaster: Establishing tcp connections between hosts behind nats. In *ACM SIGCOMM Asia Workshop*, volume 5, 2005.

[14] Jeffrey L Eppinger. Tcp connections for p2p apps: A software approach to solving the nat problem. Technical report, Technical Report CMUISRI-05-104, Carnegie Mellon University, 2005.

[15] Rohan Mahy, Philip Matthews, and Jonathan Rosenberg. Traversal using relays around nat (turn): relay extensions to session traversal utilities for nat (stun). *Internet Request for Comments*, 2010.

[16] Pinggai Yang, Jun Li, Jun Zhang, Hai Jiang, Yi Sun, and Eryk Dutkiewicz. SMBR: A novel NAT traversal mechanism for structured Peer-to-Peer communications. In *Computers and Communications (ISCC), 2010 IEEE Symposium on*, pages 535–539. IEEE, 2010.

[17] Farida Chowdhury. Structured peer-to-peer overlays for nated churn intensive networks. 2015.

[18] MaidSafe. The maidsafe platform.

[19] Bingqing Shen, Jingzhi Guo, and CL Philip Chen. A nat-ed peer organization model in kademlia protocol. In *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pages 52–59. IEEE, 2013.

[20] Petar Maymounkov and David Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 53–65, London, UK, 2002. Springer-Verlag.

[21] P. Srisuresh, G. Tsirtsis, P. Akkiraju, and A. Hefferman. Dns extensions to network address translators (dns alg). Technical report, RFC 2694, Internet Engineering Task Force, 1999.

[22] Pyda Srisuresh, Jiri Kuthan, Jonathan Rosenberg, Andrew Molitor, and Abdallah Rayhan. Middlebox communication architecture and framework, 2002.

[23] Liwei Yang and Kai Lei. Combining ice and sip protocol for nat traversal in new classification standard. In *2016 5th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 576–580. IEEE, 2016.

[24] ShengQuan Tsai. A study of p2p traversal through symmetric nat: with random port assignment. VDM Verlag Dr. Mller, 2010.

[25] Arno Wacker, Gregor Schiele, Sebastian Holzapfel, and Torben Weis. A nat traversal mechanism for peer-to-peer networks. In *2008 Eighth International Conference on Peer-to-Peer Computing*, pages 81–83. IEEE, 2008.

[26] Jonathan Rosenberg et al. Interactive connectivity establishment (ice): A protocol for network address translator (nat) traversal for offer/answer protocols. Technical report, RFC 5245, April, 2010.

[27] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '01, pages 161–172, New York, NY, USA, 2001. ACM.

[28] Ashish Patro, Yadi Ma, Fatemah Panahi, Jordan Walker, and Suman Banerjee. A system for audio signalling based nat traversal. In *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*, pages 1–10. IEEE, 2011.

[29] Gianni DAngelo and Salvatore Rampone. A nat traversal mechanism for cloud video surveillance applications using websocket. *Multimedia Tools and Applications*, 77(19):25861–25888, 2018.

[30] Sunanda Bose, Akash Chowdhury, and Nandini Mukherjee. Lepantu: Long polling based energy efficient passive nat traversal through udp. In *2019 7th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 259–264. IEEE, 2019.

Table 6. Combination of NAT Traversal Techniques.

NAT traversal techniques	**Advantages**	**Limitations**
ICE [26]	ICE is used for UDP-based media streaming.	1) It causes a long connectivity check delay 2) It requires considerable number of message exchanges 3) It may fail to create a direct communication path between two hosts if connection tracking is implemented.
Combining ICE and SIP [23]	1) It works for all types of NATs. 2) It works in peers behind the same NAT and different NATs efficiently.	It does not work for peers behind multiple levels of NAT.
Symmetric NAT Traversal with Random Port Assignment [24]	1) It can detect different network environments through standardized protocols. 2) It can traverse all types of NATs. 3) Without using relay server, it can make direct TCP connection between two hosts.	Implementation is complex.
STUN variant with Super-peers [25]	All type of NATs can be traversed.	1) Relaying is expensive as delay and overhead are added. 2) Only UDP traffics are possible.

Table 7. Traditional NAT Traversal.

NAT traversal techniques	**Advantages**	**Limitations**
Static Mapping	Simple.	1) Manual configuration. 2) Administrative privilege needed.
Port Forwarding	Can be configured manually or dynamically.	1)Expert users needed. 2) It can traverse full Cone NAT only. 3) Impractical as unable to access to ISP's router.
Relaying	1) Reliable. 2) Relatively simple.	1) Less efficient. 2) Poor QoS. 3) Dependent on a server and consumes server's processing power and network bandwidth. 4) Increased latency.
Connection Reversal	Straightforward.	1) It requires a server at the initial connection phase. 2) One of the hosts must need to have a public IP address.

Table 8. Others.

NAT traversal techniques	**Advantages**	**Limitations**
CAN [27]	1) CAN outperforms ICE in terms of DCR (Direct Communication Ratio), connectivity check delay, and resource demand. 2) CAN reduces the usage of a relay server.	1) CAN does not consider the distribution of NAT types. 2) It did not explain how to discover NAT types.
ANT [28]	1) Simple. 2) A TCP connection can be established between mobile clients behind NATs with no manual configuration. 3) No intermediate server needed.	ANT does not work well in presence of background music.
VSaas [29]	Efficient.	Dependent on a server.
LEPaNTU [30]	Energy efficient method.	Only a single communication scheme is addressed.