Network Address Translation (NAT)

Background Material for Overlay Networks Course

Jan, 2013

Prof. Sasu Tarkoma
University of Helsinki, Department of Computer Science
Contents

• Overview
• Background
• Basic Network Address Translation
• Solutions
  – STUN
  – TURN
  – ICE
• Summary
What is NAT

• Expand IP address space by deploying private address and translating them into publicly registered addresses

• Private address space (RFC 1918)
  – 10.0.0.0 - 10.255.255.255 (10.0.0.0/8)
  – 172.16.0.0 - 172.31.255.255 (172.16.0.0/12)
  – 192.168.0.0 - 192.168.255.255 (192.168.0.0/16)

• First described in RFC 1631

• Technique of rewriting IP addresses in headers and application data streams according to a defined policy

• Based on traffic source and/or destination IP address
NATs and Firewalls

• Firewalls
  – Security main concern
  – Demilitarized zone
  – Increasingly complex rules (what is filtered, how)

• NATs
  – Lightweight security devices
    • Topology hiding and firewalling
  – Increasing number in deployment
    • Solves some of the address space problems of IPv4 (Port Translation, NAPT)
  – IPv6 solves the addressing problem so NATs are not needed for this
Port Ranges

• Well-known
  – 1-1023

• Registered
  – 1024-49151

• Dynamic/Private
  – 49152-65535
NAT Example

<table>
<thead>
<tr>
<th>Inside Local IP Addr.</th>
<th>Inside Global IP Addr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.0.2</td>
<td>130.233.240.9</td>
</tr>
<tr>
<td>10.0.0.3</td>
<td>130.233.240.10</td>
</tr>
</tbody>
</table>
Two NATs

Home Network

ISP Network

Internet

10.0.0.2

192.168.0.1

130.233.240.9

10.0.0.3
Web Access Example

Connection request to port 80 from host to web server, source 10.0.0.3, port 1001

Request from host forwarded to server, source 130.233.240.9 port 2001

Response sent to 130.233.240.9 port 2001

Response forwarded to 10.0.0.3 port 1001

<table>
<thead>
<tr>
<th>Inside Local IP Addr.</th>
<th>Inside Port</th>
<th>Out IP Addr.</th>
<th>Out Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.0.0.2</td>
<td>1000</td>
<td>130.233.240.9</td>
<td>2000</td>
</tr>
<tr>
<td>10.0.0.3</td>
<td>1001</td>
<td>130.233.240.9</td>
<td>2001</td>
</tr>
</tbody>
</table>
## NAT traffic supported

<table>
<thead>
<tr>
<th>Traffic Types/Applications Supported</th>
<th>Traffic Types/Applications not Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any TCP/UDP Traffic that Does Not Carry Source and/or Destination IP Addresses in the Application Data Stream</td>
<td>IP Multicast</td>
</tr>
<tr>
<td>HTTP</td>
<td>Routing Table Updates</td>
</tr>
<tr>
<td>TFTP</td>
<td>DNS Zone Transfers</td>
</tr>
<tr>
<td>Telnet</td>
<td>BOOTP</td>
</tr>
<tr>
<td>archie</td>
<td>Talk, Ntalk</td>
</tr>
<tr>
<td>finger</td>
<td>H.323</td>
</tr>
<tr>
<td>NTP</td>
<td>VDOLive</td>
</tr>
<tr>
<td>NFS</td>
<td>NetShow</td>
</tr>
<tr>
<td>rlogin, rsh, rcp</td>
<td>VXtreme</td>
</tr>
<tr>
<td>Although the Following Traffic Types Carry IP Addresses in the Application Data Stream, they are Supported by Cisco IOS NAT:</td>
<td>SNMMP</td>
</tr>
<tr>
<td>ICMP</td>
<td></td>
</tr>
<tr>
<td>SMTP</td>
<td></td>
</tr>
<tr>
<td>FTP (Including PORT and PASV Commands)</td>
<td></td>
</tr>
<tr>
<td>NetBIOS over TCP/IP</td>
<td></td>
</tr>
<tr>
<td>Progressive Networks RealAudio</td>
<td></td>
</tr>
<tr>
<td>White Pines CuSeeMe</td>
<td></td>
</tr>
<tr>
<td>DNS &quot;A&quot; and &quot;PTR&quot; Queries</td>
<td></td>
</tr>
<tr>
<td>Xing Technologies StreamWorks</td>
<td></td>
</tr>
</tbody>
</table>
Timeouts

• It is crucial how mappings are removed
  – Done using timers

• Mapping refresh
  – Outbound refresh: internal to external
  – Inbound refresh: external to internal
    • May be used, subject to attacks

• A NAT UDP mapping must not expire in less than 2 minutes

• Established connections
  – Cisco default for TCP: 24 hours
  – Shouldn’t be less than 2 hours and 4 minutes
  – TCP sends keep alives

• NATs can have application-specific timers
NAT Features

- NAT provides transparent and bi-directional connectivity between networks having arbitrary addressing schemes
- NAT eliminates costs associated with host renumbering
- NAT conserves IP addresses
- NAT eases IP address management
- Load Balancing
- NAT enhances network privacy
- Address migration through translation
- IP masquerading
- Load balancing
NAT Concerns

• Performance
  – IP address modification, NAT boxes need to recalculate IP header checksum
  – Port number modification requires TCP checksum recalculation

• Fragmentation
  – Fragments should have the same destination

• End-to-end connectivity
  – NAT destroys universal end-to-end reachability
  – NATted hosts are often unreachable
NAT Concerns

• Applications with IP-address content
  – Need AGL (Application Level Gateway)
  – Typically applications that rely on IP addresses in payload do not work across a private-public network boundary
  – Some NATs can translate IP addresses in payload

• NAT device can be a target for attacks

• NAT behaviour is not deterministic

• NATs attempt to be transparent
  – Challenges for network troubleshooting
NAT Traversal

• Challenge: how to allow two natted hosts communicate?

• Straightforward solution: use a relay with a public address that is not natted
  – Connection reversal possible if a node has a public address
    • Relay is a rendezvous point

• More complicated solutions
  – Detect presence of NATs
  – Hole punching
Relaying

Simplest solution: traffic goes through relay
Connection reversal

B with public address uses relay to send the address to NATed node
NAT Policies I/II

• RFC3489, more recent see RFC 5389 and others.
• Old terminology defined cones, we introduce them here, but they are **problematic** and have been **superseded**.
• Full cone NAT
  – All requests from the same internal IP address and port are mapped to the same public IP address and port.
  – Once a mapping is created, all incoming traffic to the public address is routed to the internal host without checking the address of the remote host.
• Restricted cone NAT
  – Unlike a full cone NAT, a remote host (with IP address X) can send a packet to the internal host only if the internal host had previously sent a packet to IP address X.
NAT Policies II

• Port restricted cone NAT
  – A port restricted cone NAT is like a restricted cone NAT, but the restriction includes port numbers.
  – An external host can send a packet, with source IP address X and source port P, to the internal host only if the internal host had previously sent a packet to IP address X and port P.

• Symmetric NAT
  – A symmetric NAT is a NAT where all requests from the same internal IP address and port to a specific destination IP address and port are mapped to the same external source IP address and port. If the same internal host sends a packet with the same source address and port to a different destination, a different mapping is used.
  – Only the external host that receives a packet can send a UDP packet back to the internal host.
Hole Punching
Hole Punching

• Hole punching is a technique to allow traffic from/to a host behind a firewall/NAT without the collaboration of the NAT itself.

• The simplest way is to use UDP packets.
Hole Punching with Full Cone

NAT routes the packet to the rendezvous who acquires the public address ip:port. A mapping is created and a hole is punched.

Host looks up the public address of the device (NAT’s IP)

Relay with public IP address

Private IP address

Public IP address

No restrictions on IP traffic arriving at the NAT
Hole Punching with Restricted Cone

NAT routes the packet to the rendezvous who acquires the public address ip:port. A mapping is created.

Host looks up the public address of the device (NAT’s IP)

Natted host sends message to host. Hole is punched

Restricts traffic based on public IP address (not on port)

(X,y) sends to (A,z) through (N,q)

(A,w) can send back to (N,q)
Hole Punching with Port Restricted Cone

NAT routes the packet to the rendezvous who acquires the public address ip:port. A mapping is created.

Host looks up the public address of the device (NAT's IP)

Natted host sends message to host. Hole is punched

Similar as restricted cone, but source port of remote generated packed must be the same addressed by host A in the first packet.
Symmetric NATs

• Punching a hole in symmetric NAT is impossible
  – NAT assigns new mappings for different destinations
  – Random port numbers
  – Only recipient can send packet back to internal host

• Rendezvous is not useful here because NAT will assign new public address to packets for other hosts than the rendezvous

• The only way to traverse this NAT is by Connection Reversal or Relaying.
A symmetric NAT does not maintain a consistent port binding. It creates a new mapping/binding for each new session. This means that hole punching does not work, because there is no consistent opening for incoming packets.
TURN, STUN, ICE
TURN

• IETF MIDCOM draft "Traversal Using Relay NAT (TURN)" RFC 5389
• TURN is a protocol for UDP/TCP relaying behind a NAT
• Unlike STUN there is no hole punching and data are bounced to a public server called the TURN server
• TURN is the last resource. For instance behind a symmetric NAT
• It introduces a relay
  – Located in customers DMZ or Service Provider network
  – Single point of failure
  – Requires a high performance server
TURN Elements

- A TURN client is an entity that generates TURN requests (Binding request).
- A TURN Server is an entity that receives TURN requests, and sends TURN responses.
- The server is a data relay, receiving data on the address it provides to clients, and forwarding them to the clients.
- Discovery, for example using Anycast over UDP or DNS.
- Security is supported by establishing a shared secret with TLS (after discovery).
STUN

• IETF RFC 3489 "STUN – Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)"

• A client-server protocol to discover the presence and types of NAT and firewalls between them and the public Internet

• STUN allows applications to determine the public IP addresses allocated to them by the NAT

• Defines the operations and the message format needed to understand the type of NAT

• The STUN server is contacted on UDP port 3478

• The server will hint clients to perform tests on alternate IP and port numbers (STUN servers have two IP addresses)
How STUN works

• The client sends a request to the server
• The server returns a response which contains the source IP of the packet received from the client (inside the payload) i.e. the mapped IP address
• The client compares its IP address with the mapped IP address returned by the server
• If they are different then the client is behind a NAT
• The client can set some flags in the message which tell the server to send a packet from another IP/port or to another IP/port (the server has 2 public IP addresses)
STUN Limitations

- Does not work with symmetric NATs used by most corporate environments
- Does not work if both clients are behind the same NAT
- Additional deployment of a STUN server placed in the public space
- STUN NAT detection not reliable!
Additional Note

• Once a client has discovered its external addresses, it can relate it to its peers. If the NATs are full cone then either side can initiate communication. If they are restricted cone or restricted port cone both sides must start transmitting together.

• The trick is using STUN to discover the presence of NAT, and to learn and use the bindings they allocate.
Revised STUN

• Revised specification in RFC 5389
  – "Session Traversal Utilities for NAT"
  – Classical STUN not deployable
  – STUN is not a complete solution, rather it is a tool
    • ICE
    • SIP Outbound, ..
    – Binding discovery, NAT keep-alives, Short-term password, Relay (previously TURN)
• Can run on UDP, TCP, TLS
• http://tools.ietf.org/html/rfc5489
STUN Relay

• Previously TURN
• STUN server located using DNS SRV records
• Allocate request/response
  – Allocate an external address at the relay
• Send indication
  – Data to remote endpoint using relay
• Data indication
  – Data received from remote endpoints using relay
• Connect request and response
  – Request relay to establish TCP connection with remote endpoint
• Connection status indication
  – Relay informs endpoint about status of TCP connection: LISTEN, ESTABLISH, CLOSED
ICE

- IETF MMUSIC draft "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT)"
- Allows peers to discover NAT types and client capabilities
- Provide alternatives for establishing connectivity, namely STUN and TURN
- Works with all types of NATs, P2P NAT traversal
- Designed for SIP and VoIP. Can be applied to any session-oriented protocol
- The detailed operation of ICE can be broken into six steps: gathering, prioritizing, encoding, offering and answering, checking, and completing.
**ICE Step 1**

• Gathering
  – Caller gathers IP addresses and ports
  – Each is a potential candidate for communications
  – A candidate is gathered from each interface
  – The agent contacts STUN server from each host interface
  – Result is a set of server-reflexive candidates
    • IP addresses that route to the outermost NAT between the agent and the STUN server
  – Relayed candidates from TURN servers
    • IP addresses and ports on the relay servers
• Prioritizing
  – Each candidate is assigned a priority value
• Prioritizing addresses
  – Priority = $2^{24}$ (type preference) + $2^8$ (local preference) + 2 (256 – component ID)
ICE Step 3

• Encoding
  – Body of a SIP request contains SDP message
  – IP addresses and port numbers
  – ICE extends SDP by several new SDP attributes
    • Candidate attribute, used to transfer candidates (IP address, port, priority)
    • Credential information for securing STUN messaging
ICE Step 4

- Offering and Answering
  - SIP INVITE + SDP is sent to the called party
  - The called party performs the same gathering, prioritizing, and encoding that the caller performed
  - A provisional SIP response with SDP message including the candidates
ICE Step 5

• Checking
  – The caller and called party have exchanged SDP messages
  – ICE performs the work of selecting the communication parameters
  – Each agent computes a priority for the candidate pair by combining the priority of each candidate

• Verification
  – ICE uses a STUN transaction from each agent towards the other to find working candidate pairs
    • Connectivity check
  – Checks are performed sequentially (n squared!)
  – Ordered by priority, check every 20 ms
    • When receives a STUN request, generate a request in the other direction (triggered check)
ICE Step 6

- Completing
  - Once a check is complete, the agent knows that it has found a working pair
  - A final check is generated towards the other agent
  - After this final transaction is sent, the SIP phone will ring (called agent)
  - Answering will generate a SIP 200 OK message

- ICE increases connection setup delays
NATs and SIP

• Client will generate SIP INVITE and 200 OK responses with private addresses
  – In the SDP as the target for receipt of media
    • Solved by ICE (difficult)
  – In the Contact of a REGISTER
    • Solved by SIP Outbound
  – In the Via of a request
    • Solved by rport (RFC 3581)

• Recipient will not be able to send messages to the private address
  – Incoming calls do not work
  – Media will be discarded
  – Responses are not received
IPTABLES

- IPTABLES is a tool for firewall and NAT functionality
- Uses several tables and each table can have several chains
- Each table/chain can have rules that determine how packets are handled
- PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING
- NAT is done with PREROUTING, POSTROUTING
- Firewall is done with INPUT, FORWARD, OUTPUT
- Tables: nat, filter, mangle, raw
IETF BEHAVE WG

• IETF BEHAVE Working Group is working on classification of NAT behaviours
  – Behavior Engineering for Hindrance Avoidance
• Recommendations for NAT vendors
  – Towards deterministic operation
• http://www.ietf.org/html.charters/behave-charter.html
Summary

• NATs are popular on the Internet
  – Private address spaces
  – End-to-end reachability problems
• Four main types of NATs
  – Hole punching and relays basic mechanisms
• A number of solutions for NAT traversal
  – STUN
  – TURN
  – ICE
  – SIP Client-initiated Outbound Connections
Additional Notes I

• Hole Punching
  – UDP hole punching uses a public relay (typically a STUN server)
  – Address information is exchanged
  – Direct communication is tried
  – Idea: each UDP send operation opens a hole
    • Very similar to connection reversal
  – Does not work with symmetric NATs
  – Can be used with TCP as well but this case is more complex
Additional Notes II

• Difference between symmetric and port restricted cone NAT
  – The terminology is confusing (well-known issue)
  – In practice it is difficult to classify NAT behaviour
  – For port restricted the mapping is consistent
    (iAddr:port -> eAddr:port)
  – If an outside node received a packet from the internal node, it can send a packet back
  – Hole punching is possible
  – Other way to define: ingress filter on address/port

• For symmetric, only if a node receives a packet, can it send a packet back
  – The mapping changes for each destination
  – Hole punching is not possible
Additional Notes III

• Connection reversal and hole punching
  – Connection reversal can be used when one node is behind a NAT
  – Hole punching can be used even if both are behind NATs (same idea)