NUTSS: A SIP Based Approach to UDP & TCP Network Connectivity

VoIP SIP related Presentation

P93942009 左家榮
R93922115 吳瑞傑
R93942036 彭冠仁
Outline

• Introduction
• Reference
• STUN(T), TURN and ICE
• Port Prediction
• NAT TCP Solution
• Issues
• Conclusion
• Discussion
Introduction

• NAT let network connectivity between IP hosts become more complex
• SIP has been extended to establish direct UDP (ICE), and indirect TCP connection (TURN) between hosts behind NAT and firewalls
  – How about direct TCP connection?
• NUTSS: Nat, URI, TUNNEL, SIP, STUNT
• Why can’t SIP be used to establish all kinds of P2P communication?
  – Socket-like API
Reference

Outline

• Introduction
• Reference
• STUN(T), TURN and ICE
• Port Prediction
• NAT TCP Solution
• Issues
• Conclusion
• Discussion
STUN(T) overview

• STUN : Simple Traversal of UDP Through NATs
• STUNT - Simple Traversal of UDP Through NATs and TCP too
• STUN(T) allows a host to learn the global IP address and UDP(TCP) port assigned by its NAT box
• This address can be subsequently conveyed by SIP to allow direct UDP (TCP) connectivity between hosts
NAT variations

• Full Cone
• Restricted Cone
• Port Restricted Cone
• Symmetric
**Cone NAT Type**

* Full Cone
* Restricted Cone
* Port Restricted Cone

(local ip, local port)

(global ip, global port)
Symmetric NAT Type

(local ip, local port, remote ip, remote port)
STUN(T) Architecture

• STUN(T) Client
  – An entity that generate STUN(T) requests
  – Which can execute on user’s PC or network server

• STUN(T) Server
  – An entity that receive STUN(T) requests and sends STUN(T) responses
  – Generally attached to the public internet
STUN(T) Binding Request and Response
TURN overview

- TURN: Traversal Using Relay NAT
- TURN allows a host to select a globally-addressable TCP relay
- It can subsequently be used to bridge a TCP connection between two NATed hosts
- TURN does not allow direct connectivity between NATed hosts
  - Indirect TCP connection!
TURN Architecture

Addr: X
Addr: T
Addr: Y
Addr: Z

Nat Server

Client 1, Registered id: S
Client 2
Other host

Addr: X ←→ PubAddr: S
S and Y setup Connection

Reject!
TURN Introduction

• Every user process can only keep an inbound link
  – For safety purpose
• May cause heavy load on TURN server
ICE overview

• ICE: Interactive Connectivity Establishment
• A Methodology for NAT Traversal for Multimedia Session Establishment Protocols
• Use STUN and TURN without extensions
• Use SDP as an option
Outline

- Introduction
- Reference
- STUN(T), TURN and ICE
- Port Prediction
- NAT TCP Solution
- Issues
- Conclusion
- Discussion
Cone NAT Type Port Prediction
Symmetric NAT Type Port Prediction

* Most NAT use uniform increment on GP#
* We can do nothing on random mapping!

(local ip, local port, remote ip, remote port)
Issues with Port Prediction

• We don’t want an interruption while determining what kind of NAT it’s behind

• There is a gap in time between the probe flow to the STUNT box and the subsequent flow to the remote host
  – The gap depends on whether we know the GP and GA of the remote host or not
Issues with Port Prediction

<table>
<thead>
<tr>
<th>Host</th>
<th>Nat</th>
<th>STUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open socket</td>
<td>Probe (s)</td>
<td></td>
</tr>
<tr>
<td>Mapping</td>
<td>Predicted mapping</td>
<td></td>
</tr>
<tr>
<td>Remote host's mapping</td>
<td>TCP/UDP Packet</td>
<td></td>
</tr>
<tr>
<td>Re-open socket</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

gap
Likelihood of Failure

• Survey several NAT boxes on the market
  – Netgear, Linksys, Dlink, Hawking, Speedstream

• Consider the majority type of NAT that customers will choose
  – Home users → simple NAT Boxes
    • No random port assignment machines
    • Very few port interrupt problems
  – Enterprise users
    • Connection fail results from their manager’s policy!
Outline

• Introduction
• Reference
• STUN(T), TURN and ICE
• Port Prediction
• NAT TCP Solution
• Issues
• Conclusion
• Discussion
NAT TCP Possible Solutions

• Candidate solution:
  – Allow users to explicitly make a new port mapping of firewall rule
    • But you should define the protocol first and make sure that NAT Box producers will take your advice!
  – Simultaneous SYN packet sending by TCP protocol specification
    • But MS windows does not support
Simultaneous Open in TCP

Use four packets to establish the connection

From W. RICHARD STEVENS : UNIX Network Programming Volu1
Simultaneous Open in TCP on Linux

<table>
<thead>
<tr>
<th>No.</th>
<th>Size</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>74</td>
<td>0.000000</td>
<td>192.168.124.2</td>
<td>192.168.124.1</td>
<td>TCP</td>
<td>5134 &gt; 5134 [SYN] Seq=0 Ack=0 win=5840 Le</td>
</tr>
<tr>
<td>2</td>
<td>74</td>
<td>0.004239</td>
<td>192.168.124.1</td>
<td>192.168.124.2</td>
<td>TCP</td>
<td>5134 &gt; 5134 [SYN] Seq=0 Ack=0 win=5840 Le</td>
</tr>
<tr>
<td>3</td>
<td>74</td>
<td>0.004312</td>
<td>192.168.124.2</td>
<td>192.168.124.1</td>
<td>TCP</td>
<td>5134 &gt; 5134 [SYN, ACK] Seq=0 Ack=1 win=23</td>
</tr>
<tr>
<td>4</td>
<td>66</td>
<td>0.004335</td>
<td>192.168.124.1</td>
<td>192.168.124.2</td>
<td>TCP</td>
<td>5134 &gt; 5134 [ACK] Seq=1 Ack=1 win=5840 Le</td>
</tr>
<tr>
<td>5</td>
<td>66</td>
<td>10.005143</td>
<td>192.168.124.2</td>
<td>192.168.124.1</td>
<td>TCP</td>
<td>5134 &gt; 5134 [FIN, ACK] Seq=1 Ack=1 win=58</td>
</tr>
<tr>
<td>6</td>
<td>66</td>
<td>10.005525</td>
<td>192.168.124.1</td>
<td>192.168.124.2</td>
<td>TCP</td>
<td>5134 &gt; 5134 [ACK] Seq=1 Ack=2 win=5840 Le</td>
</tr>
<tr>
<td>7</td>
<td>66</td>
<td>10.005579</td>
<td>192.168.124.1</td>
<td>192.168.124.2</td>
<td>TCP</td>
<td>5134 &gt; 5134 [FIN, ACK] Seq=1 Ack=2 win=58</td>
</tr>
</tbody>
</table>
New Strategy

• Consider the author’s new strategy:
  – Both tcp endpoints send SYN with very short TTL to enable the predicted ports
  – Use SIP network to send port end SYN information to each other
  – Use STUNT server to generate spoofed SYNACK
  – Client sends normal ACK back to establish the connection
New Strategy

Intent to connect

Accept

Port prediction

Port prediction

Global mapping

Global mapping

SYN (low TTL) #seq=S1

SYN (low TTL) #seq=S2

Encapsulated SYN

Encapsulated SYN

Encapsulated SYNACK

Encapsulated SYNACK

SYNACK (spoofed)

SYNACK (spoofed)

#seq=S2, #ack=S1+1

#seq=S1, #ack=S2+1

TCP ACK #ack=S2+1

TCP ACK #ack=S1+1
A New Solution: Negotiate Stage

• Sender & receiver make sure that a connection will set up
• Each endpoint ask STUNT server for next-connection-port
• Exchange the port information (GP) to each other
A New Solution: TCP setup

- Each end point send a SYN packet with very short TTL
- keep the SYN that just have sent out
- Both endpoints send the SYN stored at previous step to the other via TURN server
- Send the SYN packets (one form its raw socket and one from the other) to STUNT server and the server returns spoofed SYNACK
- Client sends normal ACK back to establish the connection
Outline

• Introduction
• Reference
• STUN(T), TURN and ICE
• Port Prediction
• NAT TCP Solution
• Issues
• Conclusion
• Discussion
NAT Characteristics

• NAT may release the port mapping in response to an ICMP TTL Exceed message
• The hop count between two NATs can’t be too small
  – Drop silently
  – Return ICMP error
  – TCP RST packet
Other Requirements

• We may need super-user permission to create raw sockets

• Different OSs have different responses to an ICMP error
  – Block unwanted ICMP error message

• The STUNT server must have the ability to spoof packets from arbitrary IPs
TCP Connection Setup

Client A | NAT N | TURN | STUNT | NAT M | Client B

Intent to connect

Accept

Port prediction

Port prediction

Global mapping

Global mapping

SYN (low TTL)

SYN (low TTL)

Encapsulated SYN

Encapsulated SYN

Encapsulated SYNACK

Encapsulated SYNACK

Encapsulated SYNACK

Encapsulated SYNACK

SYNACK (spoofed)

SYNACK (spoofed)

TCP ACK

TCP ACK
A less general approach

Intent to connect

Accept

Port prediction

Port prediction

Global mapping

Global mapping

SYN (low TTL)

Close the socket and reopen for listening

Ready to receive connection

TCP SYN

TCP SYNACK

TCP ACK
Other Requirements

• A less general approach need not to spoof packets
  – NAT N may complain about the incoming SYN
  – Abort the connection at A may result in a RST which would close the mapping in NAT N
Outline

• Introduction
• Reference
• STUN(T), TURN and ICE
• Port Prediction
• NAT TCP Solution
• Issues
• Conclusion
• Discussion
Conclusion

• This paper presents a technique which expands the scope of connectivity establishment through NATs

• Suggest that it’s appropriate to expand ICE/STUN/TURN to include all data communications between P2P users
Discussion

• Firewall and proxy server are both problems
  – Restricted remote open port number 80
  – HTTP only
  – Proxy

• Skype’s behavior under firewall and NAT
  – Skype has solved the above problems
  – Skype permits the super node behave as a relay server like a TURN server