Characterization and Measurement of TCP Traversal Through NATs and Firewalls

Saikat Guha, Paul Francis

Cornell University

IMC 2005
P2P connectivity through NATs

New inbound flows cannot be routed
P2P connectivity through NATs

New inbound flows cannot be routed
P2P connectivity through NATs

I am Dan

I am Bob

Basic solution for UDP

Saikat Guha

TCP Traversal Through NATs
P2P connectivity through NATs

1.1.1.1 2.1.1.1

Dan Bob

Bob is 2.1.1.1:2

Dan is 1.1.1.1:1

1.1.1.1 2.1.1.1

10.1.1.1 10.1.1.2

Dan Bob

Basic solution for UDP

Saikat Guha TCP Traversal Through NATs
P2P connectivity through NATs

Basic solution for UDP

Saikat Guha  TCP Traversal Through NATs
P2P connectivity through NATs

Basic solution for UDP

Saikat Guha
TCP Traversal Through NATs
P2P connectivity through NATs

Basic solution for UDP

Saikat Guha

TCP Traversal Through NATs
P2P connectivity through NATs

TCP establishment more complex

Dan 1.1.1.1 10.1.1.1

Bob 2.1.1.1 10.1.1.2

SYN

2.1.1.1

Saikat Guha
TCP Traversal Through NATs
P2P connectivity through NATs

TCP establishment more complex

Saikat Guha  TCP Traversal Through NATs
Context for this work

- **'92**: NAT Invented
- **'93-'96**: NAT traversal presumed impossible
- **'97-'01**: UDP traversal solved and standardized [Kegel]
- **'93-'03**: TCP traversal presumed impossible
- **'04**: TCP traversal ‘solved’ (2 approaches) [Guha]
- **'05**: 2 more approaches [Ford, Biggadike]
- **'05**: Approaches evaluated [Guha]
- **'06**: TCP traversal standardized
Context for this work

NAT Invented
UDP traversal solved and standardized [Kegel]
NAT traversal presumed impossible

TCP traversal presumed impossible
TCP traversal ‘solved’ (2 approaches) [Guha]
2 more approaches [Ford, Biggadike]

Many trade-offs
- NAT sensitivity
- Ease of Implementation
- Ease of Deployment

4 approaches

Saikat Guha TCP Traversal Through NATs
Context for this work

Contributions:
- Characterization
- Measurements
- Guidelines
- Standardization

NAT Invented UDP traversal solved and standardized [Kegel]
NAT traversal presumed impossible
TCP traversal presumed impossible
TCP traversal 'solved' (2 approaches) [Guha]
2 more approaches [Ford, Biggadike]
TCP traversal standardized
Approaches evaluated [Guha]

Saikat Guha
TCP Traversal Through NATs
“Take away” Results

- TCP can be established between NAT’ed peers
- Works an estimated 85%–90% of the time today
- 100% for certain popular, well-behaved NATs
  - All NATs could standardize to this
P2P TCP Establishment

Use Rendezvous Service

Saikat Guha  TCP Traversal Through NATs
Use Rendezvous Service

P2P TCP Establishment

1.1.1.1 10.1.1.1

2.1.1.1 10.1.1.2

Dan

10.1.1.1

Bob

10.1.1.2

??

SYN

Dan

2.1.1.1

Bob

10.1.1.1

TCP Traversal Through NATs
P2P TCP Establishment

Punch hole using connect/close/bind/listen.

Dan NAT SYN ?? NAT Bob

time

Saikat Guha TCP Traversal Through NATs
P2P TCP Establishment

Punch hole using connect/close/bind/listen

Saikat Guha  TCP Traversal Through NATs
P2P TCP Establishment

Accept incoming connection

Dan → NAT
- close()
- bind()
- listen()

NAT → ??
- SYN

Bob

TCP Traversal Through NATs
P2P TCP Establishment

Accept incoming connection

Saikat Guha  TCP Traversal Through NATs
P2P TCP Establishment

What if: NAT returns RST, closes hole
What if: NAT rejects SYN through hole
P2P TCP Establishment

Variation: low-TTL SYN
P2P TCP Establishment

Variation: low-TTL SYN, spoof SYNACK
Variation: low-TTL SYN, RAW SYNACK
P2P TCP Establishment

What if: NAT blocks outgoing SYNACK
Recap

4 approaches
  - 16 variants (mix and match)

Many trade-offs
  - Some sensitive to NATs behavior
  - Some hard to implement
  - Some hard to deploy

Measurement study to determine how well each works in practice
Methodology

- Implemented all approaches
  - Lessons learned in the paper
- Cause of failure for 16 brands of NATs
  - Linksys, DLink, Netgear, Belkin, ...
- 32 axis of classification
- Classified (∼100) NATs in the wild
- Extrapolated for world-wide behavior
  - Brand share market analysis
## NAT Axes of Classification

### NAT Binding:
- **Type**
  - Delta
  - Hairpin
- **Overloading**
  - Max Flows
  - Predictable

### Preservation:
- **Port Number**
  - Low
  - High
- **Dynamic Parity**
  - Low
  - High
  - Sequential

### Packet Mangling:
- **TCP Data**
- **ICMP Data**
- **TCP Sequence**

### Filters:
- \(\text{SYN}\)
- \(\text{SYN} \rightarrow \text{SYN}\)
- \(\text{SYN} \rightarrow \text{ICMP2} \rightarrow \text{SYN}\)
- \(\text{SYN} \rightarrow \text{ICMP11} \rightarrow \text{SYNACK}\)
- \(\text{SYN} \rightarrow \text{SYNACK}\)
- \(\text{SYN} \rightarrow \text{RST} \rightarrow \text{SYNACK}\)
- \(\text{SYN} \rightarrow \text{ICMP11} \rightarrow \text{SYN}\)
- \(\text{SYN} \rightarrow \text{ICMP2} \rightarrow \text{SYNACK}\)
- \(\text{SYN} \rightarrow \text{SYNACK}\)

### Timers:
- **SYN-SENT**
- **Established**
- **Timed-Wait**

---

Saikat Guha  
TCP Traversal Through NATs
### NAT Axes of Classification

<table>
<thead>
<tr>
<th>NAT Binding:</th>
<th>Type</th>
<th>Overloading</th>
<th>Delta</th>
<th>Max Flows</th>
<th>Hairpin</th>
<th>Hairpin</th>
<th>Predictable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preservation:</td>
<td>Port Number</td>
<td>Low</td>
<td>High</td>
<td>Parity</td>
<td>High</td>
<td>Sequential</td>
<td></td>
</tr>
<tr>
<td>Packet Mangling:</td>
<td>TCP Data</td>
<td>ICMP Data</td>
<td>TCP Sequence</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Filters:</td>
<td>SYN</td>
<td>SYN (known IP)</td>
<td>Estd. SYN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYN</td>
<td>SYN</td>
<td>SYN RST SYN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYN ICMP2 SYN</td>
<td>SYN SYNACK</td>
<td>SYN RST SYNACK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SYN ICMP11 SYNACK</td>
<td>SYN ICMP2 SYNACK</td>
<td>SYN SYNACK</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timers:</td>
<td>SYN-SENT</td>
<td>Established</td>
<td>Timed-Wait</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Port Prediction

Problem: What port did SYN come from?
Port Prediction

Dan
Port: 1037

NAT
Port: 6501

Classification

NB: Independent
Port Prediction

Dan

Port: 1037

NAT

Port: 6501

Classification

NB: Independent

Saikat Guha

TCP Traversal Through NATs
Port Prediction

Dan

Port: 1037

NAT

Port: 6501

predicted: 6501
to Bob

Classification

NB: Independent
Port Prediction

TCP Traversal Through NATs

Classification

NB: Delta
Port Prediction

Dan

Port: 1037

NAT

Port: 6501
6502
6503
6504
6505

to Bob
predicted: 6505

Classification

NB:Delta

Saikat Guha  TCP Traversal Through NATs
Port Prediction

Dan

Port: 1037

NAT

Port: 6501
6502
6503
6504
6505
6506

wrongly predicted: 6505

to Bob

Classification

NB:Delta

Saikat Guha  TCP Traversal Through NATs
Port Prediction

TCP Traversal Through NATs

Classification

NB: Random

Port: 1024
Port: 9516
6364
5289
8172
Projected Success

TCP traversal succeeds 85%-90% (estd.)
1. **STUNT Spoof** – Hard to deploy
2. **STUNT Plain** – Best Option
3. **NATBlaster** – Fails on WinXP SP2
4. **P2PNAT** – Fails on WinXP and earlier
- NAT Traversal Library
  - JAVA implementation available
  - Encrypted tunnel application
- NAT Classification software
  - Windows, Linux versions available
Future Work

- Wide-scale testing
  - Implement in bittorrent, swarmcast, ...
- Standardize NAT TCP Behavior
  - IETF BEHAVE Working Group
  - I-D: draft-hoffman-behave
Related Issues

IPv6 . . .
- Transition will require v4–v6 NATs

Firewalls . . .
- Will persist even with IPv6

Universal Plug-and-Play (UPnP) . . .
- Off by default
TCP NAT Traversal works!
  - 85%-90% today, 100% soon

For P2P developers:
  - Application guidelines
  - TCP traversal library

For NAT vendors:
  - Standards document
  - NAT checking software

http://nutss.net/stunt