TCP Hole Punching
Based on SYN Injection

Sebastian Holzapfel, Matthäus Wander, Arno Wacker, Torben Weis

August 27, 2011
Motivation

- Network Address Translation
  - Outgoing packet sets up mapping
- IPv6
  - Not yet widely deployed
- Configure port forwarding
  - Manual interaction
- UPnP (or RSIP, MIDCOM, NAT–PMP, …)
  - Limited availability
  - UPnP test with random NAT users: 4/40 [Holzapfel2011]
- 2700 BitTorrent users: 48% reachable [Jünemann2011]
Hole Punching

- Establish connectivity by sending packets mutually
- In general easier with UDP than with TCP
- Application requiring reliable transport
  - Reliable transport over UDP (RUDP)
  - TCP Hole Punching
System Model

- Out-of-band communication channel
  - Rendezvous server
  - Open P2P node

- Determine external IP endpoints used by $R_A$, $R_B$
  - Which map onto a connection between $(A, R_B)$ and $(B, R_A)$
  - E.g. STUNT [Guha2004] or MFB [Holzapfel2011]
Goal and Idea

- Use TCP sockets
- Work in practical environments
- Avoid unusual communication flow
Approach

1. Set low TTL, connect(), capture SYN, send OOB request
2. Receive OOB, send SYN out
3. Inject to listen()
4. Capture ACK, reset TTL, resend ACK
Prerequisites

P1: Raw socket

A

R_A

SYN

R_A endpoint

seq#

SYN

inject SYN

B

R_B

SYN

ACK

TTL exceeded

seq#

SYN

TTL exceeded

ACK

TTL exceeded

www.vs.uni-due.de

Matthäus Wander
Prerequisites

P1: Raw socket
P2: No seq# rewrite
Prerequisites

- P1: Raw socket
- P2: No seq# rewrite
- P3: Unaffected by TTL exceeded

Diagram:
- A sends SYN
- R_A endpoint
- TTL exceeded
- SEQ# injected SYN
- SYN/ACK received
- R_A endpoint
- TTL exceeded
- ACK received
- TTL exceeded
- ACK sent
Prerequisites

P1: Raw socket
P2: No seq# rewrite
P3: Unaffected by TTL exceeded
P4: SYN out, SYN/ACK out, ACK in
Evaluation

- Testbed with 12 NAT routers
  - 10 embedded devices, 2 OS distributions
Evaluation

- Testbed with 12 NAT routers
  - 10 embedded devices, 2 OS distributions

- Factory defaults
- 132 test cases:
  - Set up default route
  - Determine IP endpoint
  - Run SYNI
  - Exchange payload
- Analyze network trace
## Results

<table>
<thead>
<tr>
<th>A/B</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>2</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>5</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>6</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>7</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>8</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>9</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>10</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>11</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>12</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>
Results – P4

P1: Raw socket
P2: No seq# rewrite
P3: Unaffected by TTL exceeded
P4: SYN out, SYN/ACK out, ACK in
Conclusion

- TCP Hole Punching
  - Out-of-band SYN transport
  - Local SYN injection

- Evaluated 132 router combinations
  - 56 successful test cases
  - 16 timeouts: improve with reversal
  - 60 n/a: improve endpoint determination

- Very different router behavior
  - Analyze network traces to evaluate effectiveness
  - Set of mechanisms with different prerequisites
S. Guha et al.: „STUNT – Simple Traversal of UDP Through NATs and TCP too”, December 2004
References (2/2)

- S. Guha et al.: „NUTSS: A SIP–based Approach to UDP and TCP Network Connectivity“, 2004
- A. Biggadike et al.: „NATBLASTER: Establishing TCP Connections Between Hosts Behind NATs“, April 2005