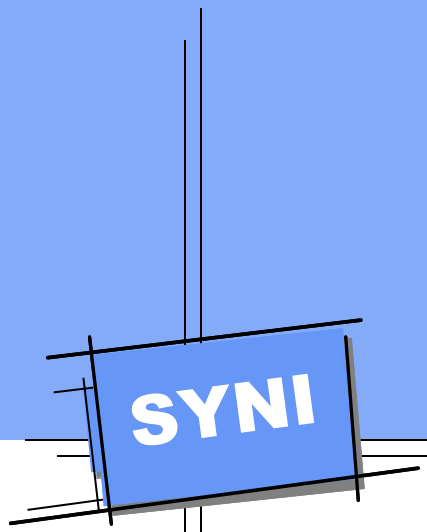
TCP Hole Punching Based on SYN Injection



Sebastian Holzapfel, Matthäus Wander,
Arno Wacker, Torben Weis

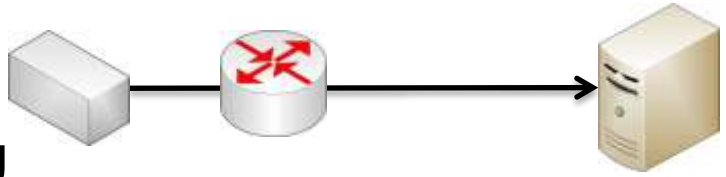
August 27, 2011

UNIVERSITÄT DUISBURG ESSEN

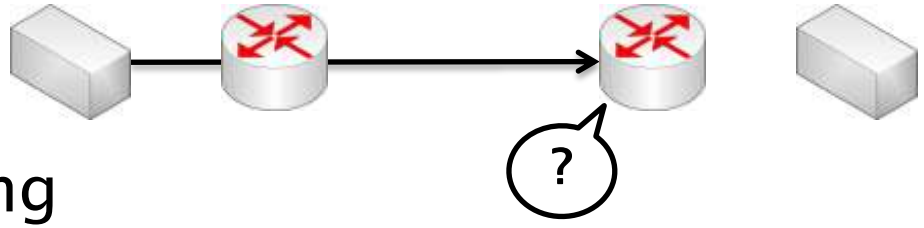
www.vs.uni-due.de

Motivation

- Network Address Translation
  - Outgoing packet sets up mapping



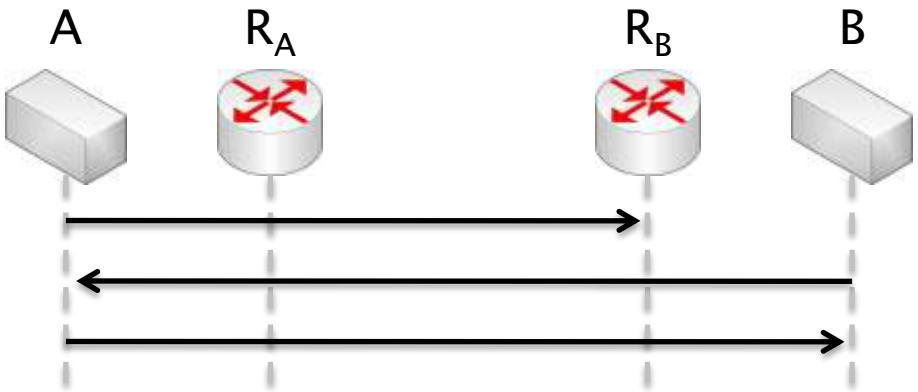
- IPv6
  - Not yet widely deployed



- 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

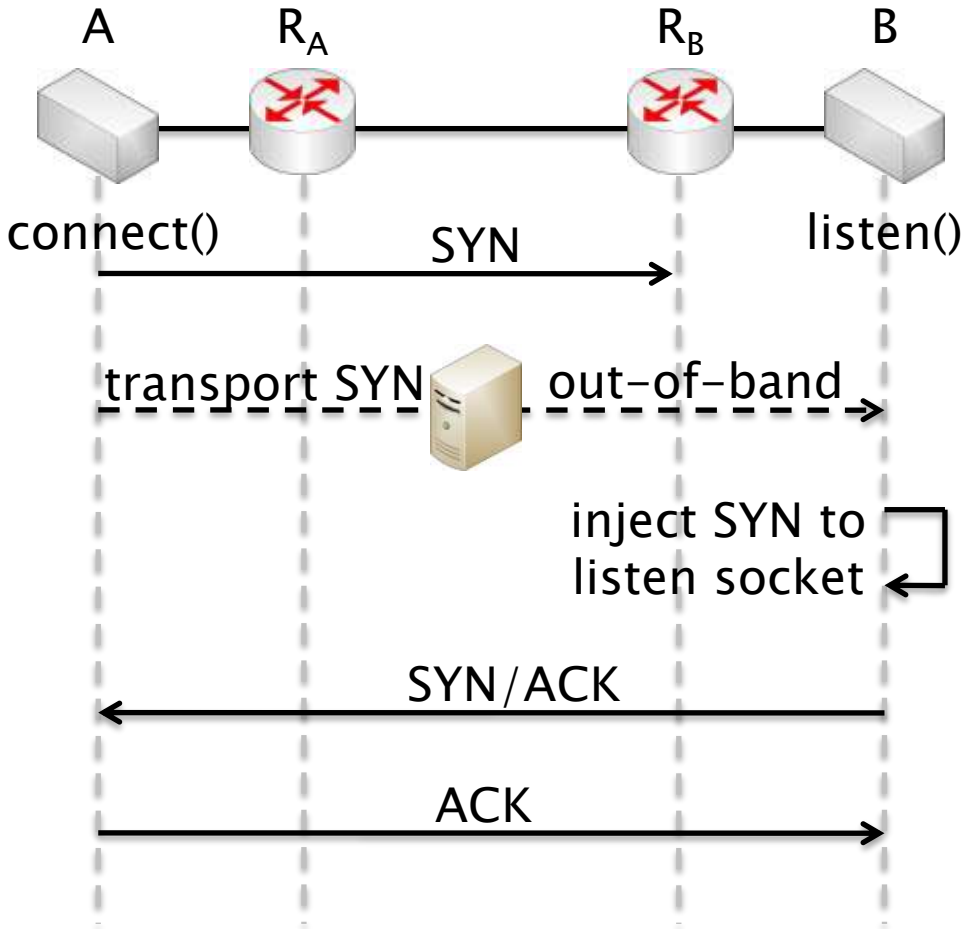


- In general easier with UDP than with TCP
- Application requiring reliable transport
  - Reliable transport over UDP (RUDP)
  - TCP Hole Punching

System Model

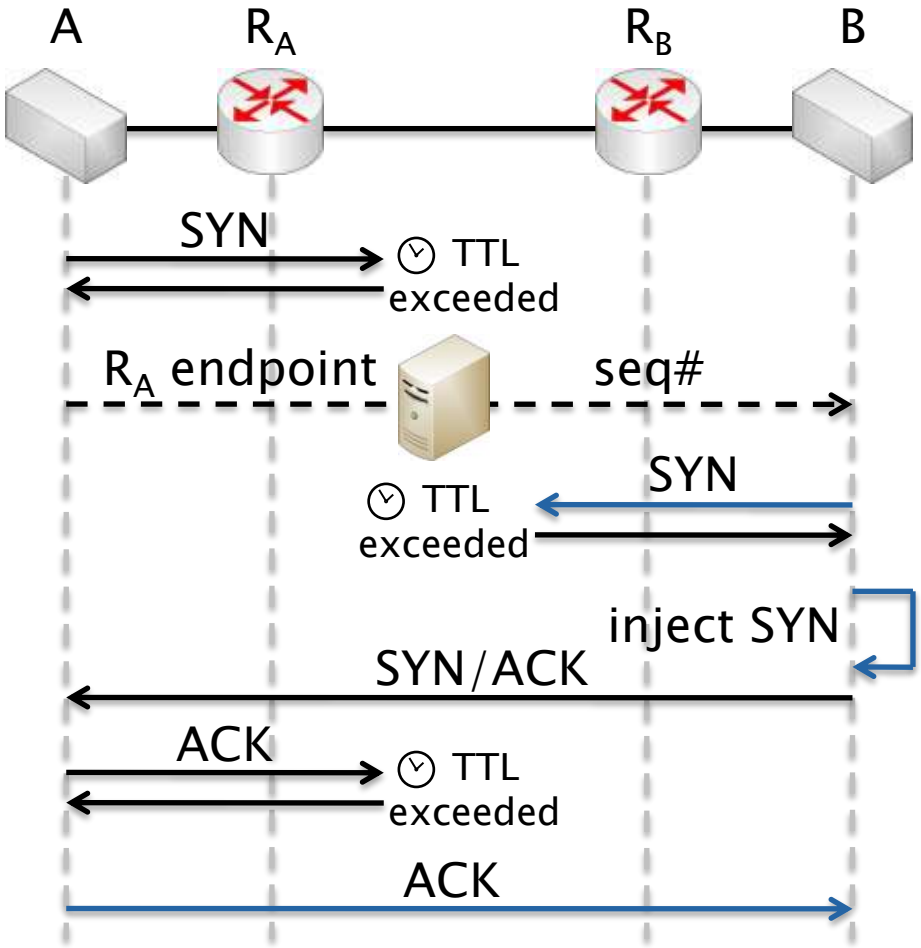
- Out-of-band communication channel
  - Rendezvous server
  - Open P2P node
- Determine external IP endpoints used by R_A, R_B
  - Which map onto a connection between (A, R_B) and (B, R_A)
  - E.g. STUNT [Guha2004] or MFB [Holzapfel2011]

Goal and Idea



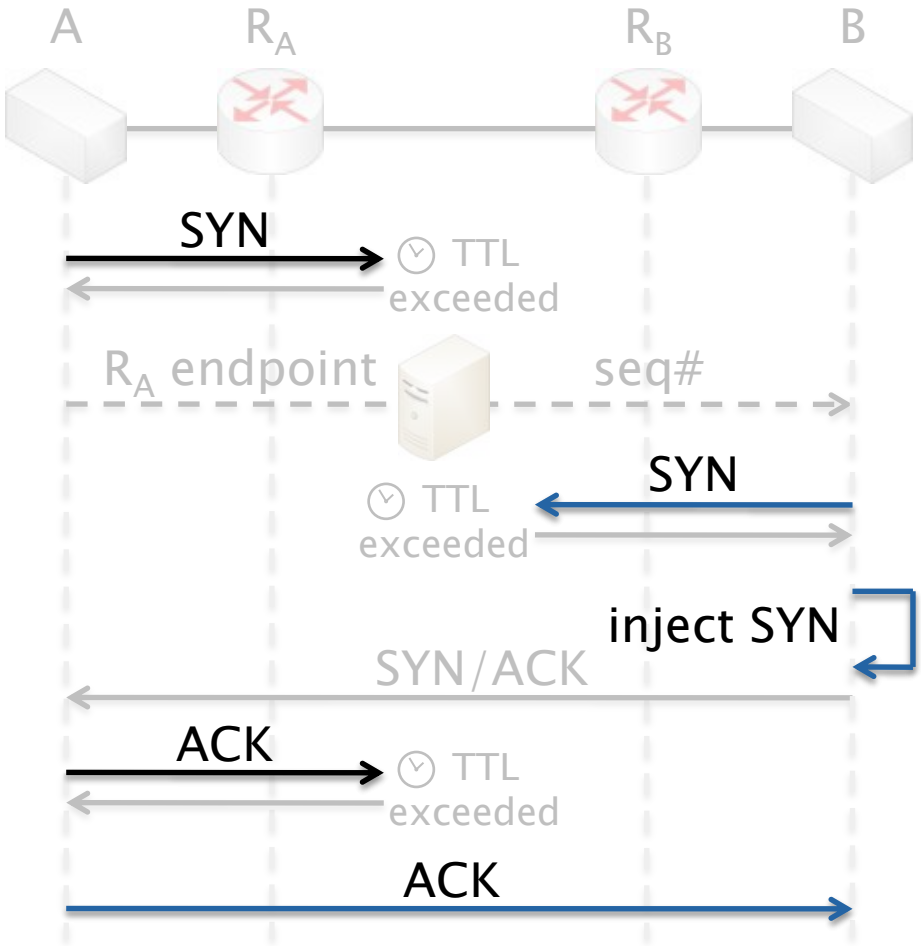
- Use TCP sockets
- Work in practical environments
- Avoid unusual communication flow

Approach



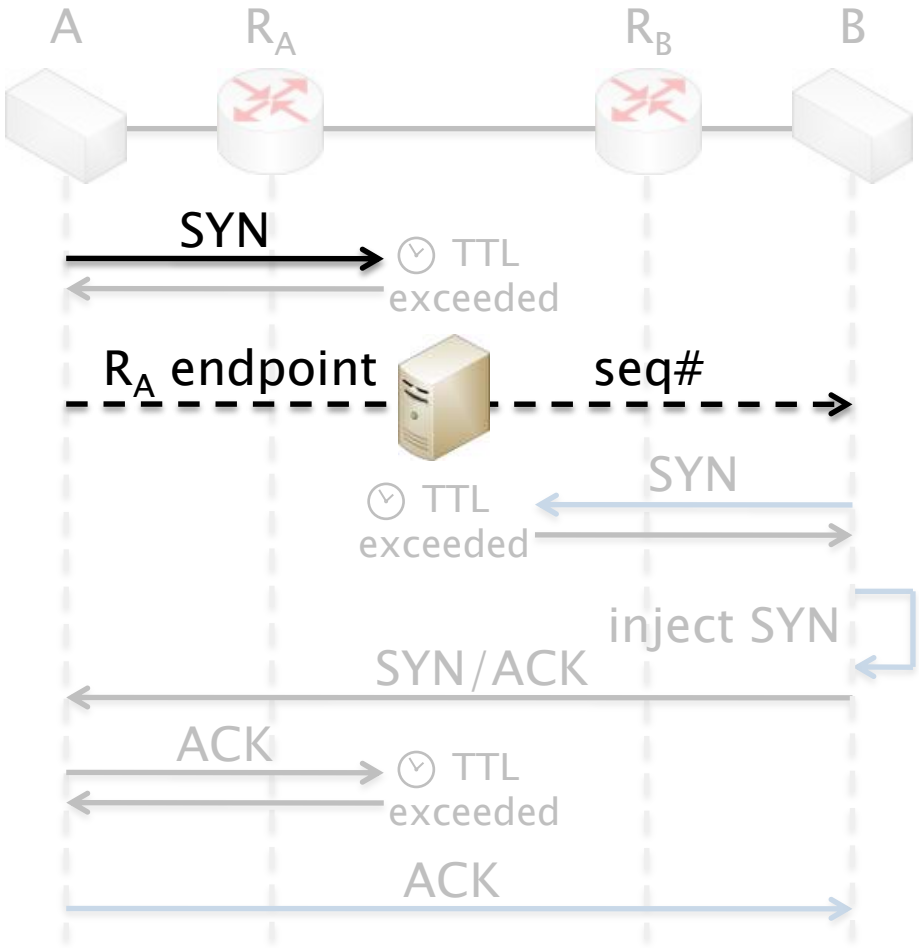
- 1. Set low TTL, connect(), capture SYN, send OOB request
- 2. Receive OOB, send SYN out
- 3. Inject to listen()
- 4. Capture ACK, reset TTL, resend ACK

Prerequisites



P1: Raw socket

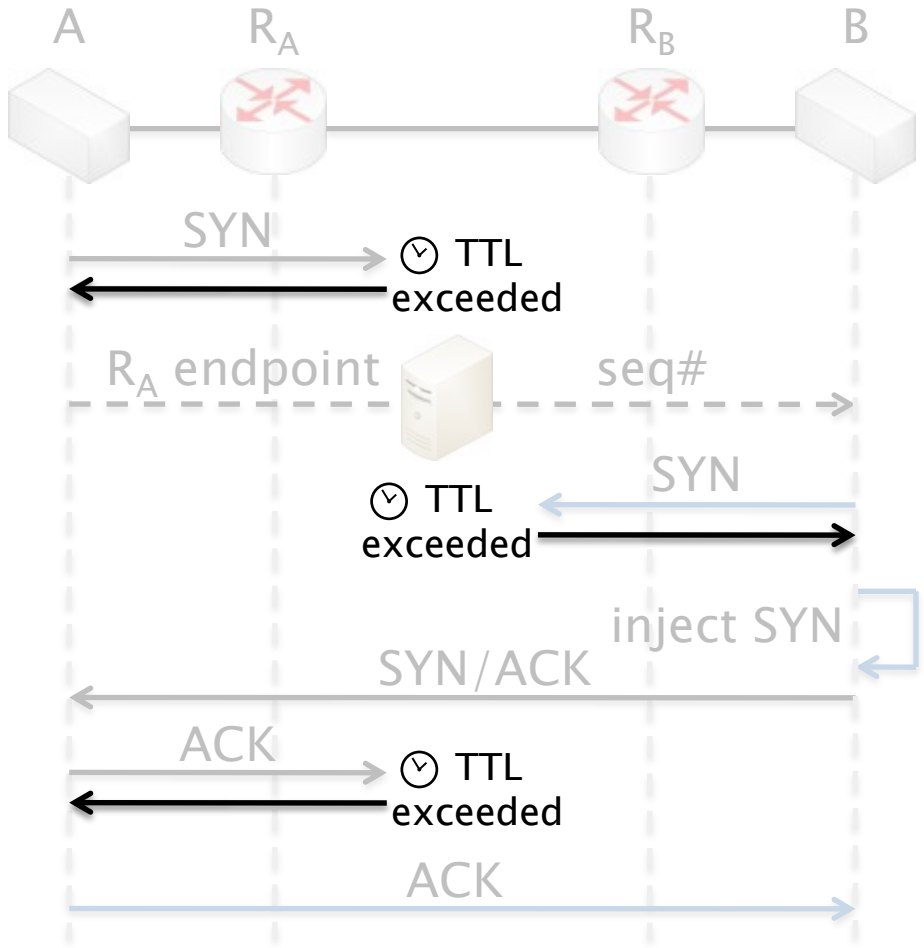
Prerequisites



P1: Raw socket

P2: No seq# rewrite

Prerequisites

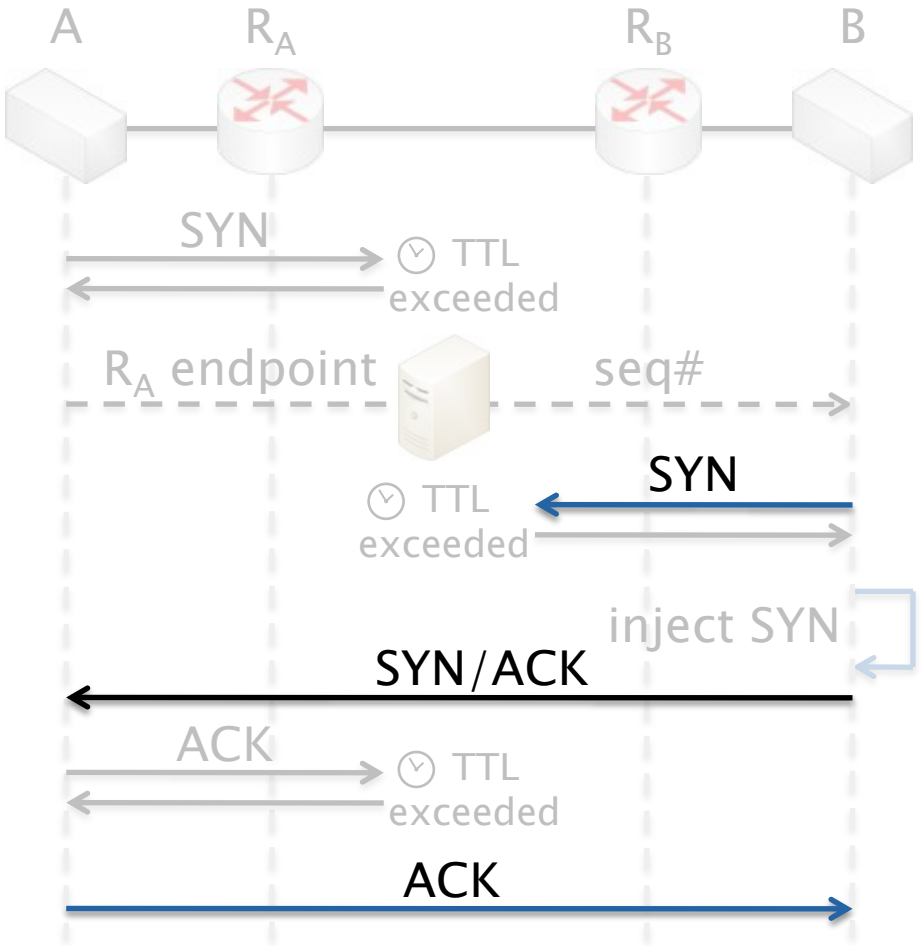


P1: Raw socket

P2: No seq# rewrite

P3: Unaffected by TTL exceeded

Prerequisites



P1: Raw socket

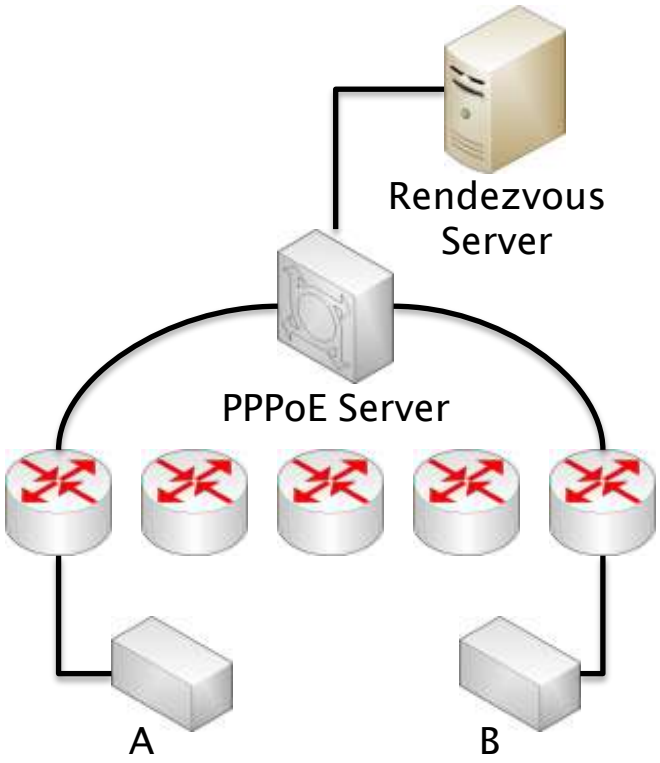
P2: No seq# rewrite

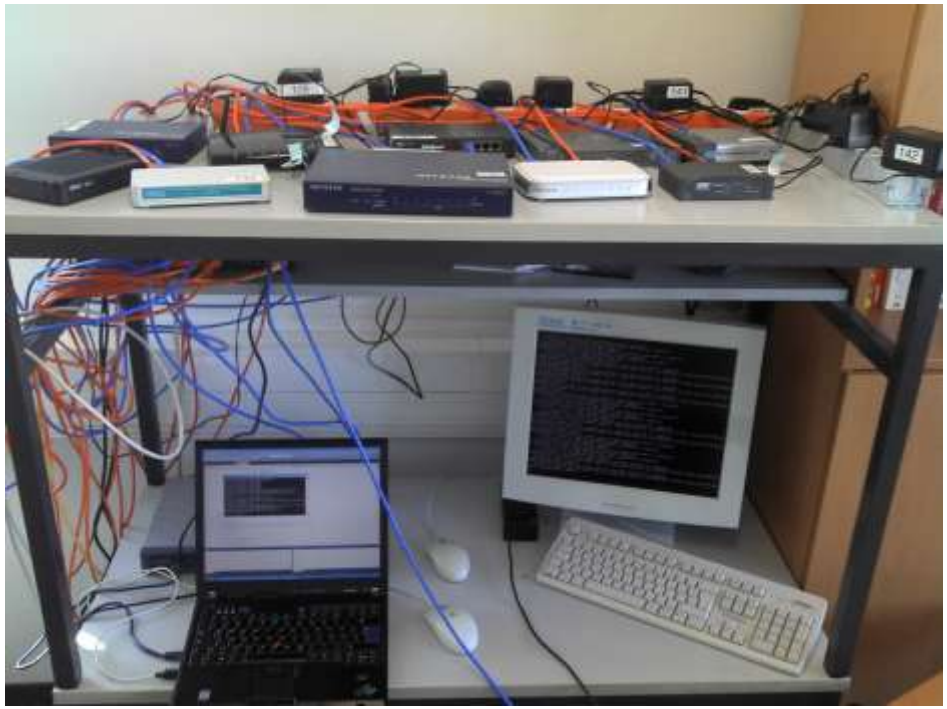
P3: Unaffected by TTL exceeded

P4: SYN out, SYN/ACK out, ACK in

Evaluation

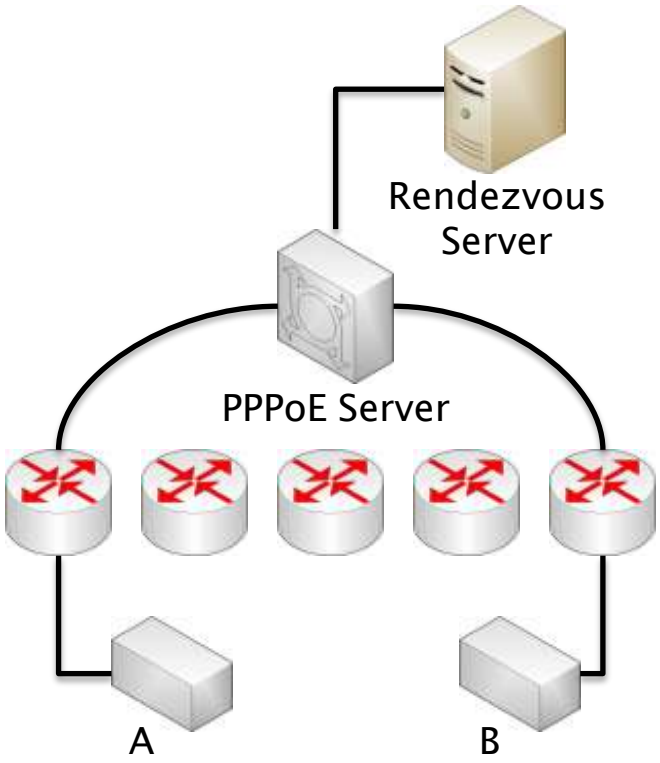
- Testbed with 12 NAT routers
  - 10 embedded devices, 2 OS distributions





Evaluation

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

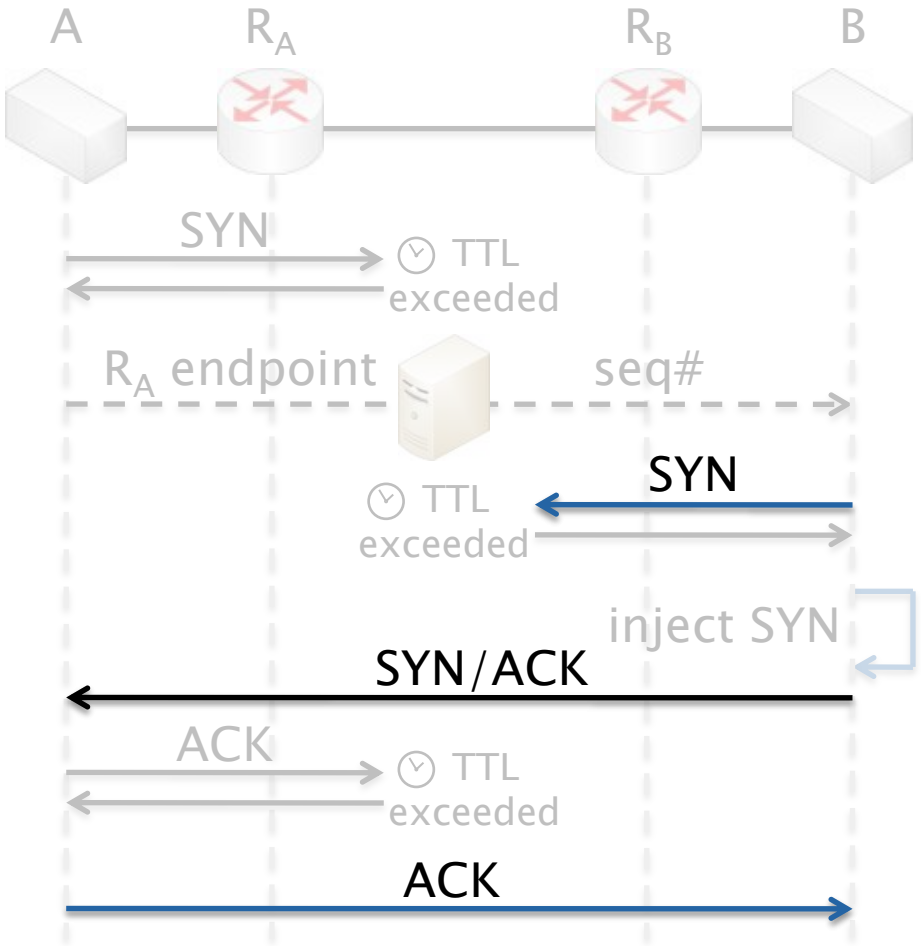


- Factory defaults
- 132 test cases:
  - Set up default route
  - Determine IP endpoint
  - Run SYNI
  - Exchange payload
- Analyze network trace

Results

A/B	1	2	3	4	5	6	7	8	9	10	11	12
1		×	✓	✓	×	×	✓	✓	×	✓	✓	×
2	×		×	×	×	×	×	×	×	×	×	×
3	✓	×		✓	×	×	✓	✓	×	✓	✓	×
4	✓	×	✓		×	×	✓	✓	×	✓	✓	×
5	✓	×	✓	✓		×	✓	✓	×	✓	✓	×
6	✓	×	✓	✓	×		✓	✓	×	✓	✓	×
7	✓	×	✓	✓	×	×		✓	×	✓	✓	×
8	✓	×	✓	✓	×	×	✓		×	✓	✓	×
9	×	×	×	×	×	×	×	×		×	×	×
10	✓	×	✓	✓	×	×	✓	✓	×		✓	×
11	✓	×	✓	✓	×	×	✓	✓	×	✓		×
12	×	×	×	×	×	×	×	×	×	×	×	

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

References (1 /2)

- S. Holzapfel et al.: "**A New Protocol to Determine the NAT Characteristics of a Host**", May 2011
- K. Jünemann et al.: „**Towards a Basic DHT Service: Analyzing Network Characteristics of a Widely Deployed DHT**", August 2011
- 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
- B. Ford et al.: „**Peer-to-Peer Communication Across Network Address Translation**“, 2005
- A. Biggadike et al.: „**NATBLASTER: Establishing TCP Connections Between Hosts Behind NATs**“, April 2005