Discovering IPv6-in-IPv4 Tunnels in the Internet

Lorenzo Colitti,
Giuseppe Di Battista,
Maurizio Patrignani
Roma Tre University

Outline

- Network discovery in tunneled networks
- Tunneling in IPv6
- Tunnel discovery methods
- Current state of tunnels in the IPv6 Internet
- Security considerations
- Conclusions

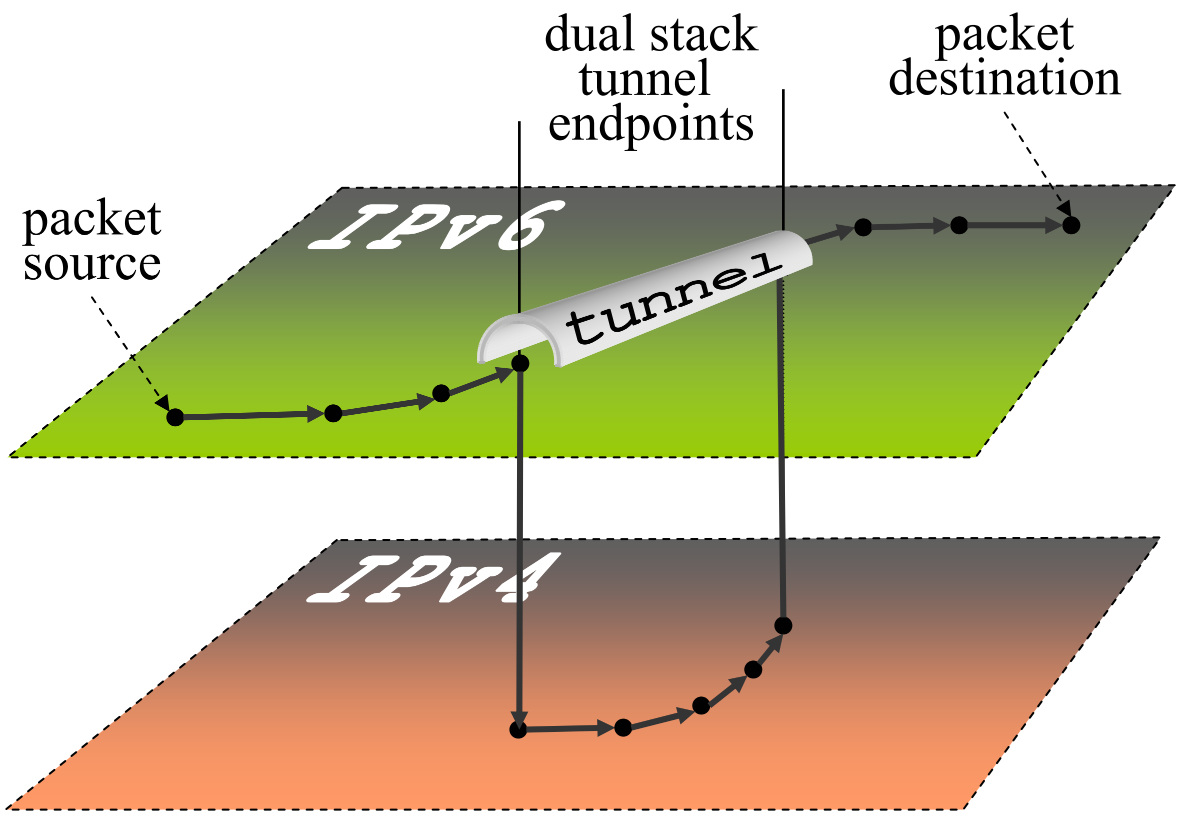
Exploration and visualization of the Internet

- Purposes
  - Fault isolation
  - Performance evaluation and planning
  - Simulation
  - Efficient deployment of network services
- Why we want to perform it automatically
  - Network complexity
  - Network size
  - Distributed administrative responsibility
  - Dynamic environment

Exploration of tunneled networks

- A tunneled network is made up of two separate layer 3 topologies that interact
- Resulting network is a complex “overlay” of two forwarding planes
- Applying known methods to explore each plane separately is not enough
  - To do this would mean to ignore the path taken by tunneled packets in the encapsulating network

Tunnel detection is necessary



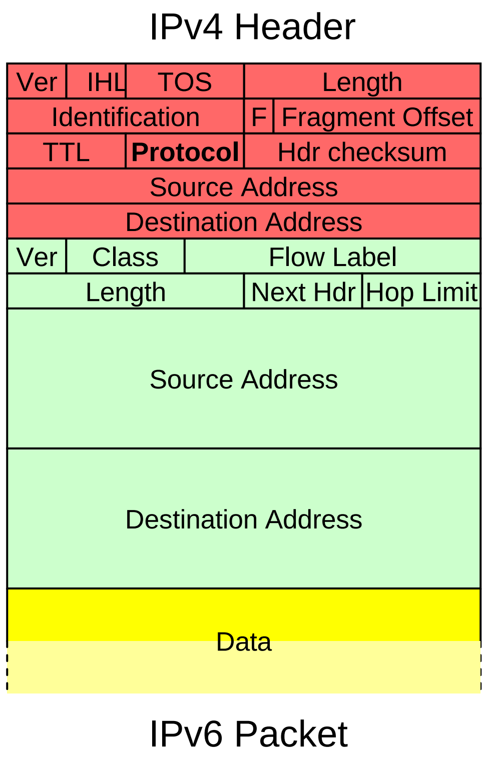
The path taken by the packet depends on both the IPv6 and IPv4 forwarding planes!

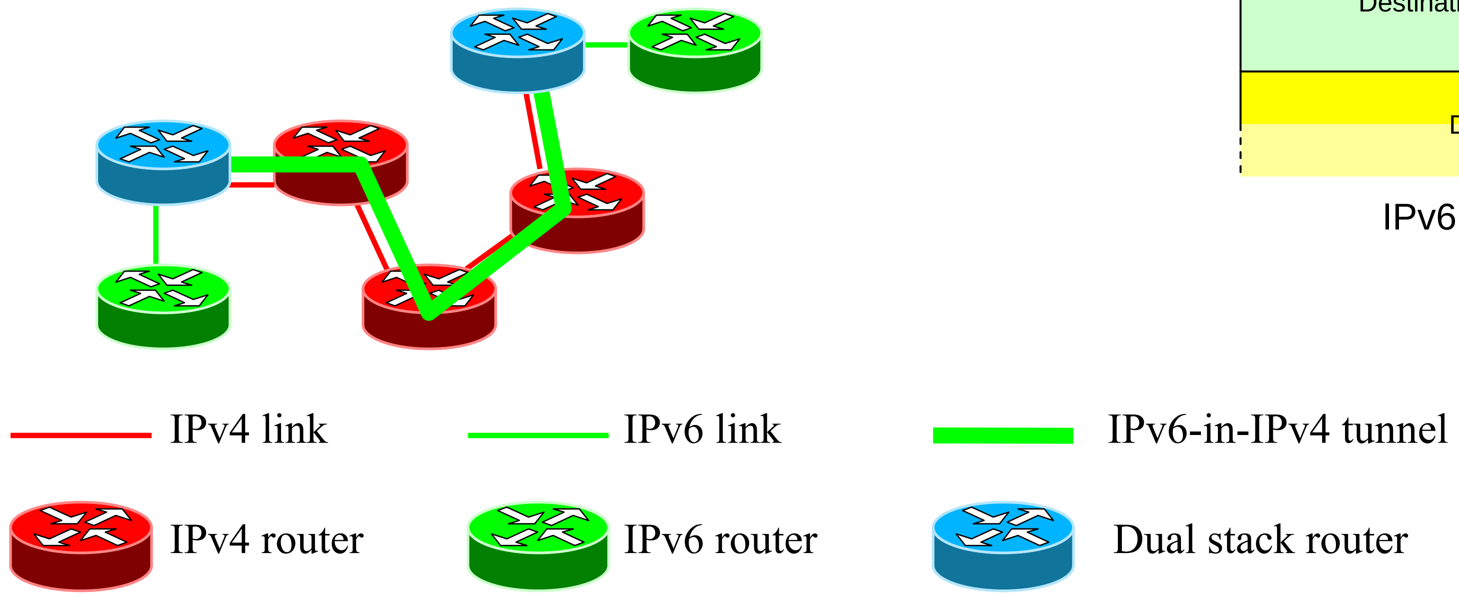
Transition to IPv6 heavily relies on tunnels

- (Manually) Configured tunnels
- Tunnel Broker
  - To dynamically create configured tunnels
- Automatic tunnels
  - One end-point is the destination host
- 6to4 Tunnels
  - To connect 6to4 sites
- ISATAP
  - “Intra-Site Automatic Tunnel Addressing Protocol”
- Teredo
  - “Tunneling IPv6 over UDP Through NATs”
- …

What is an IPv6-in-IPv4 tunnel?

- Point-to-point link between two routers
- IPv6 uses IPv4 as its "link layer"
- IPv6 packets are encapsulated in raw IPv4 packets (Protocol = 41)
- Tunnel MTU ≤ IPv4 MTU - 20





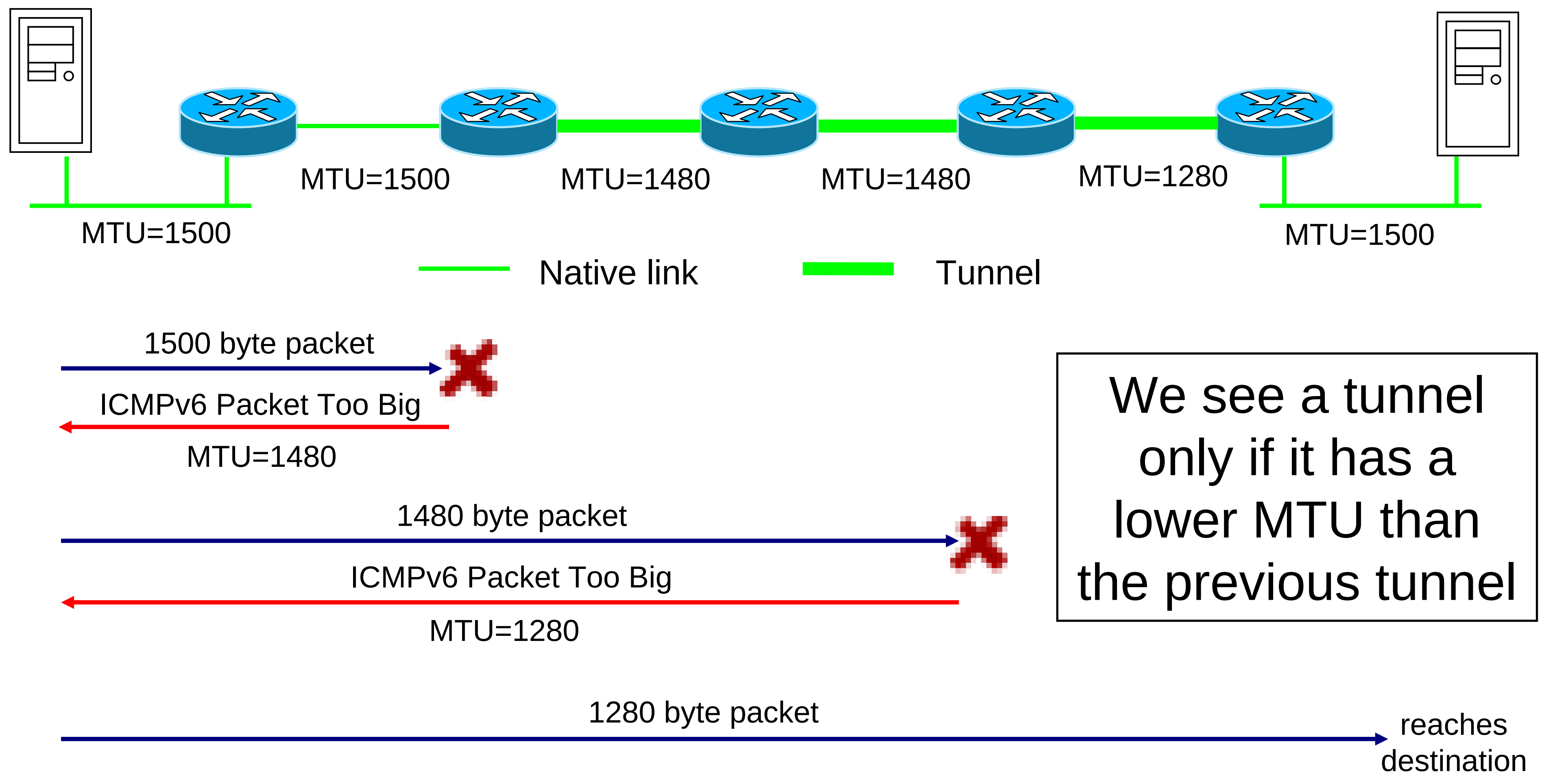
Problems with tunnels

- Low performance
  - Heavy on routers
  - Encourage inefficient routing
- Difficult to troubleshoot
- Pose security problems
- To avoid them we must know they're there
  - Transparent to IPv6, "single-hop"
    - Traceroute doesn't see them
  - What can we do?
  - (What we can't do: DNS)

Tunnel discovery rules

- MTU
- (DNS)
- Packet injection
- Injected ping
- Fragment injection
- Dying packet
- Ping-pong packet
- Bouncing packet

MTU rule

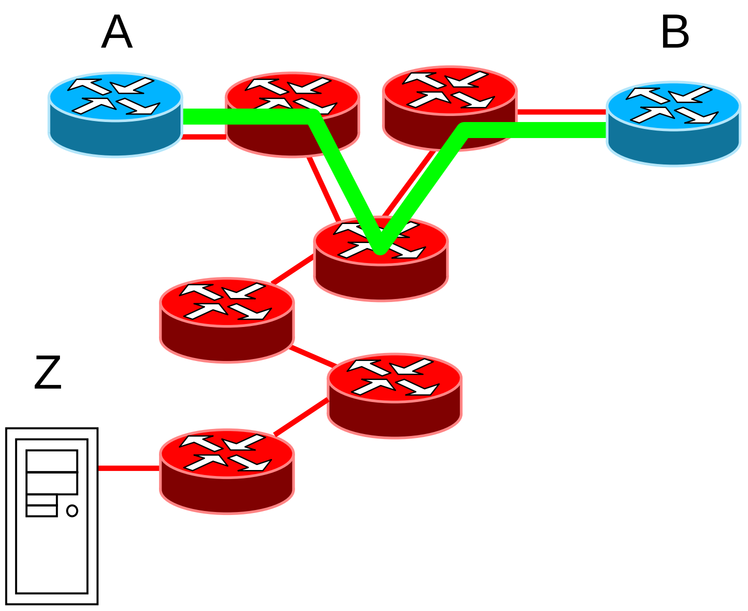


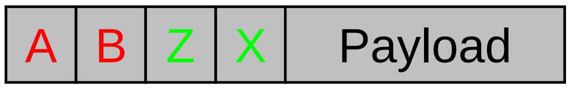
We see a tunnel only if it has a lower MTU than the previous tunnel

$$PMTU(i)<PMTU(i-1) \wedge PMTU(i) \in \{1480,1476,1472,1280\} \Rightarrow Tunnel(A(i);B(i))$$

Packet injection rule (1)

- Tunnels provide no authentication mechanism
- If Z knows the IPv4 endpoints of the tunnel, it can source IPv6 packets from B
  - Z spoofs A's IPv4 address and sends an encapsulated packet to B
  - B thinks the packet is from A
  - So it decapsulates the IPv6 packet and processes it normally
- As if Z had a direct L2 link to B



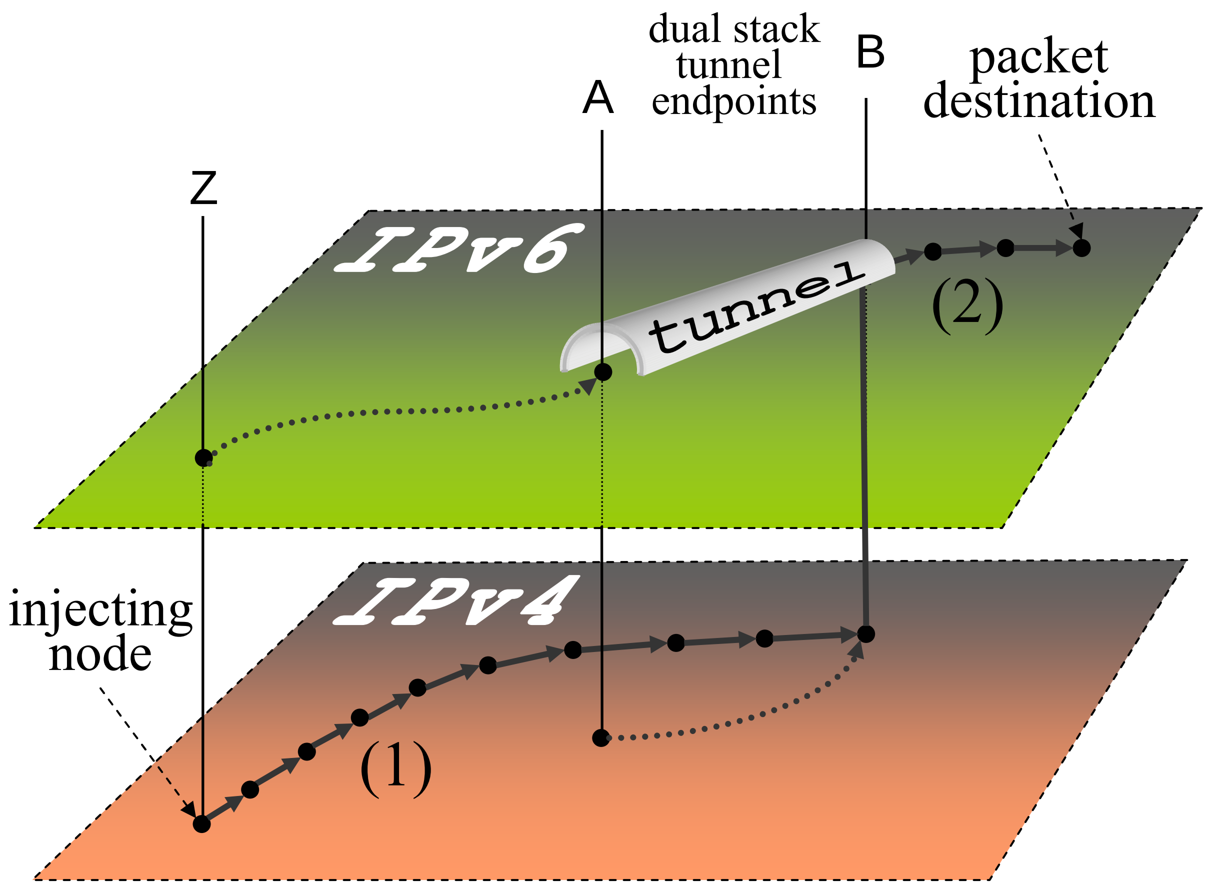


Encapsulated IPv6 packet

A = IPv4 address of A

A = IPv6 address of A

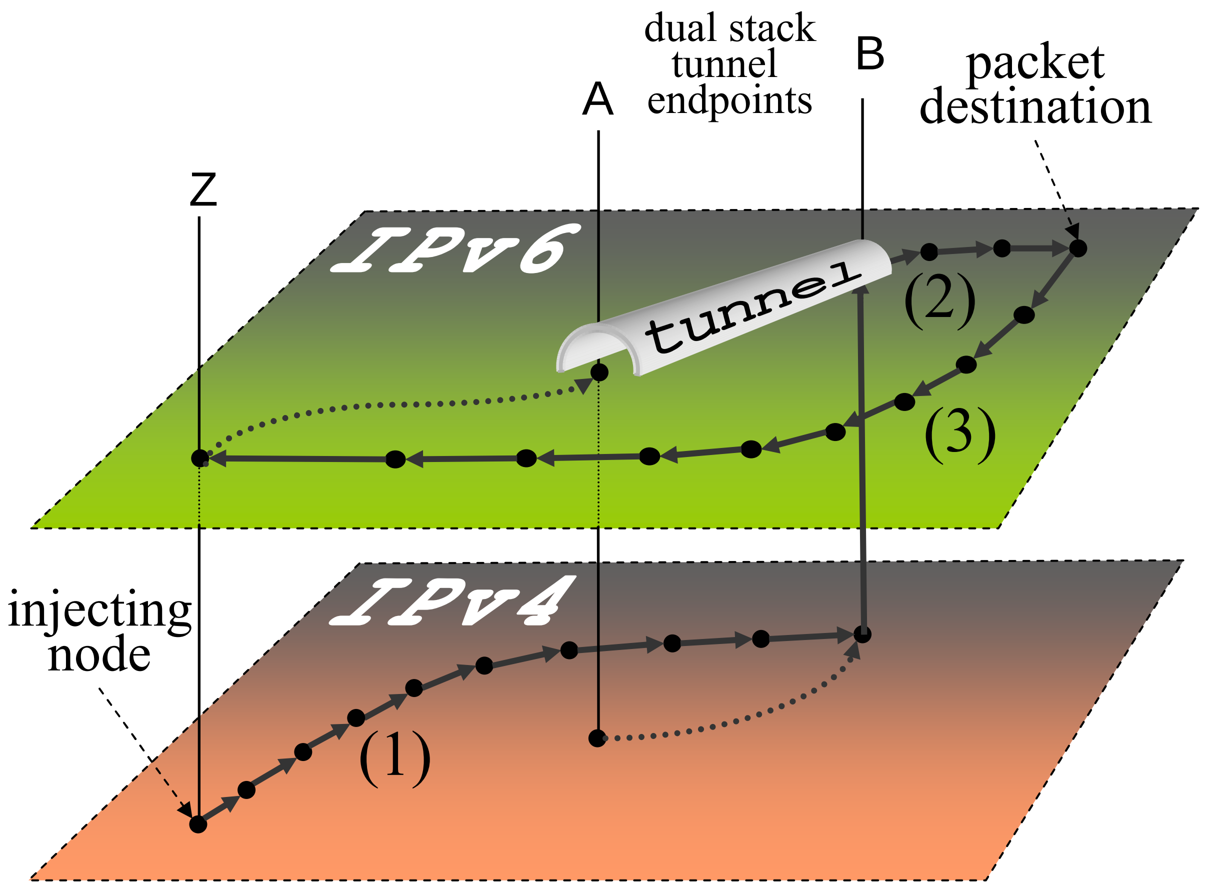
Packet injection rule (2)



$\text{Tunnel}(A;B) \Rightarrow Z{:}[A_4B_4[X_6Y_6 \text{ payload}]]$ ▲ $[X_6Y_6 \text{ payload}]{:}B$

Injected ping rule

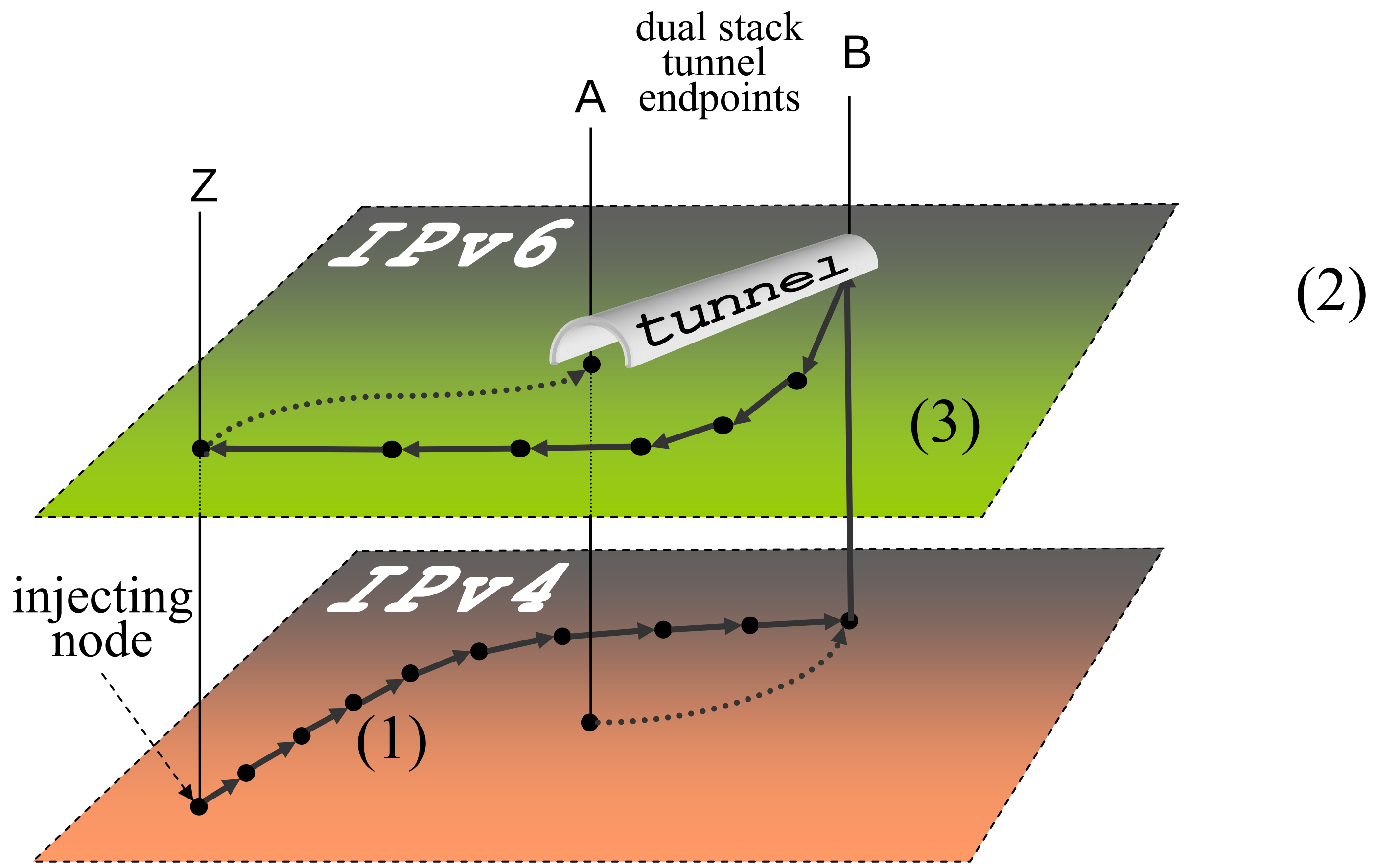
to confirm the presence of a tunnel



$Z{:}[A_4B_4[Z_6Y_6 \text{ echo-request}]] \blacktriangle [Y_6Z_6 \text{ echo-reply}]{:}Z \Rightarrow \text{Tunnel}(A;B)$

Dying packet rule

to find the IPv6 address of a tunnel endpoint



$Z{:}[A_4B_4[Z_6X_6\ HL = 1]] \searrow [Y_6Z_6\ \text{time-exceeded}]{:}Z \Rightarrow B_6 = Z_6$

Ping-pong rule

- Discover the IPv6 addresses of the endpoints
- Send hop limited ping-pong packets

$$Z{:}[A_4B_4[Z_6X_6 \text{ echo-request,HL=2}]] \leadsto [X_6Z_6 \text{ echo-reply}]{:}Z \Rightarrow A_6 = X_6$$
$$Z{:}[A_4B_4[Z_6X_6 \text{ echo-request,HL=2}]] \leadsto [Y_6Z_6 \text{ time exceeded}]{:}Z \Rightarrow A_6 = Y_6$$

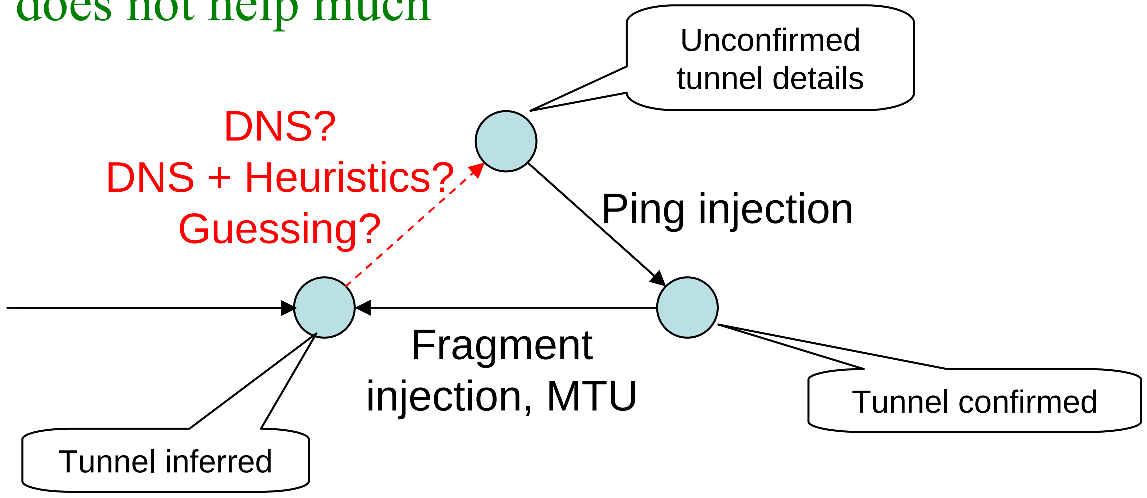
where

$$X_6 = B_6 + 1 \qquad \text{if } B_6 \text{ is even}$$
$$X_6 = B_6 - 1 \qquad \text{if } B_6 \text{ is odd}$$

- Bouncing packet rule: similar, but using source routing instead of ping-pong

Fragment injection rule
to find more tunnels from B

- Find more tunnels from B
  - IPv6 packet size ≤ MTU of tunnel
  - But IPv4 packets can be fragmented
- A tunnel is a *vantage point* from which Z can explore the rest of the network, scaling up the discovery process
- The problem is obtaining the IPv4 addresses of the endpoints
  - DNS does not help much



State of tunnels in the Internet

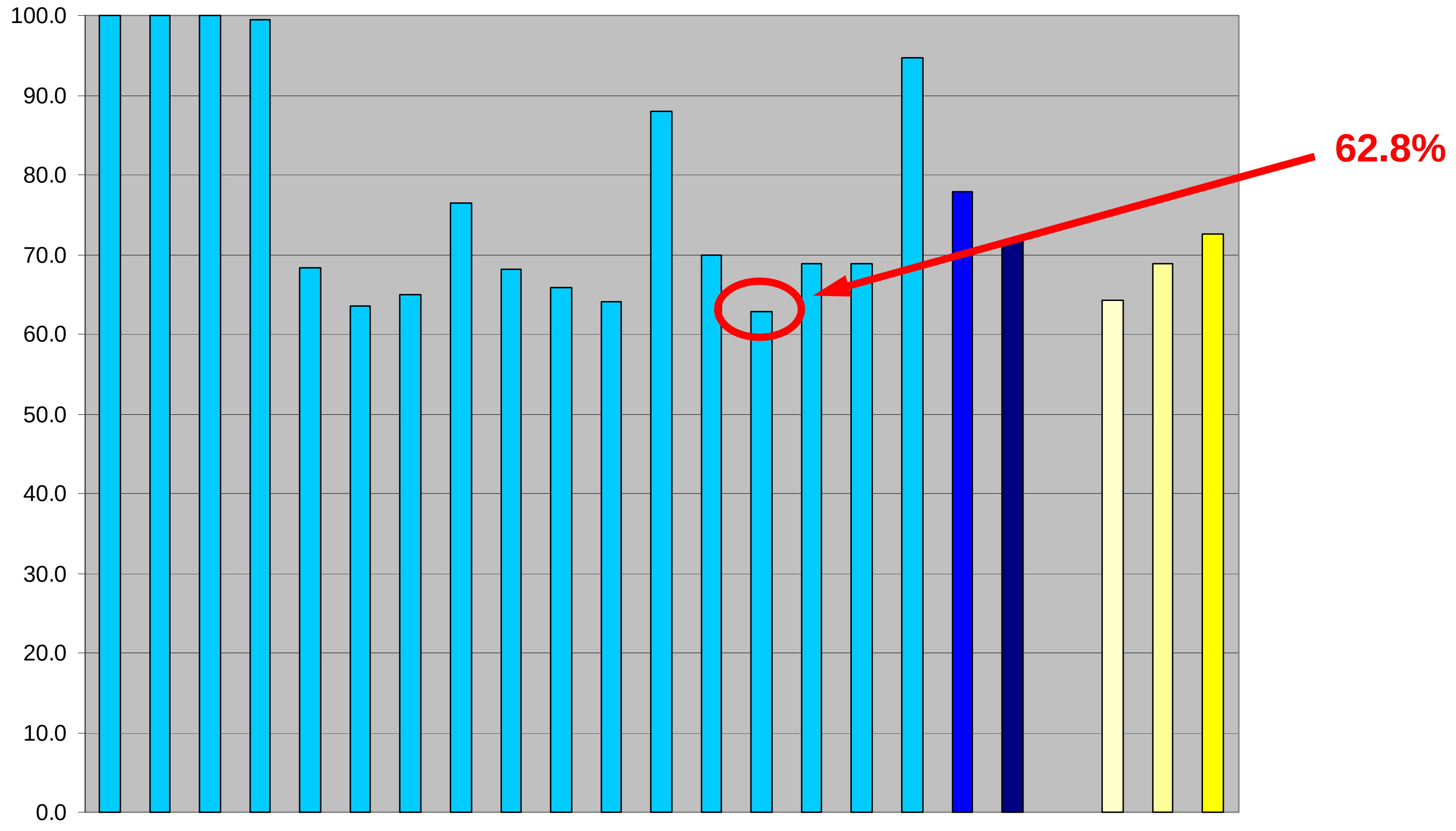
- We can measure from:
  - Tunnels in the 6bone registry
    - Over 4000 tunnels
      - ~43% nonexistent, ~32% down or filtered
    - **~1000 vantage points**
      - Mostly in tunneled networks
  - IPv6-enabled RIPE NCC TTM test-boxes
    - **~ 20 vantage points**
      - Mostly in native networks
  - Selected native IPv6 networks
    - AS137, AS3333, AS2500
- Basic idea: find MTU from each vantage point to all prefixes in BGP table

Tunnels seen from the 6bone

- Experiment done in Aug 2003
- Scan all prefixes from all vantage points, aggregate values
- Result: tunnels dominant
  - Cisco/Linux (1480) and BSD (1280) about the same
  - GRE is much less common
- **Only 8% of paths are native**
  - These vantage points are biased towards tunnels as they are themselves tunnels
  - What about native networks?

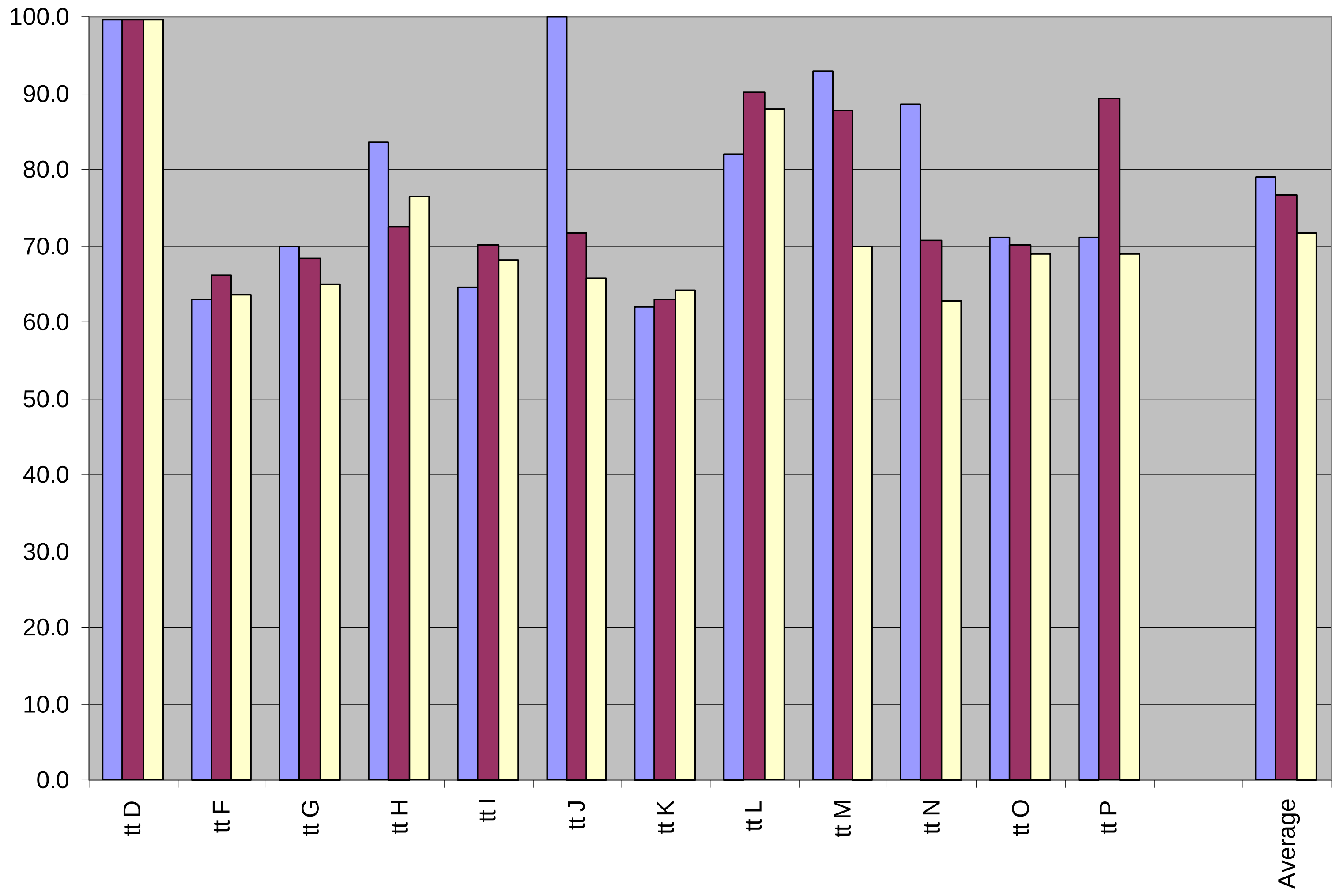
MTU	# paths	%
1480	150946	39.4
1280	138358	36.1
1476	44404	11.6
1500	31525	8.2
1428	13619	3.6
Other	4104	1.1
Total	382956	100.0

Tunnels seen from native networks



🟦	TT boxes	🟦	TT average	🟦	TT avg (native)
🟨	GARR	🟨	RIPE NCC	🟨	WIDE

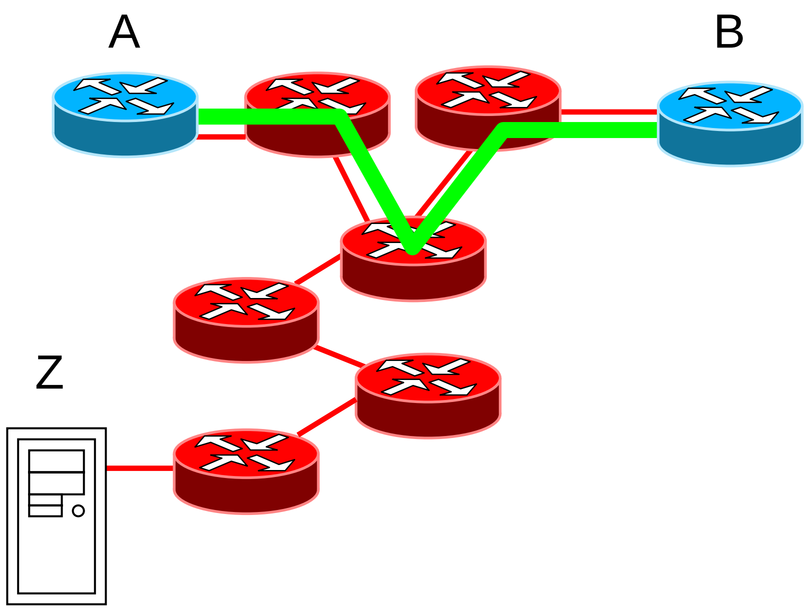
Evolution of tunnels seen by TTM



🟦 Aug 2003 🟪 Jan 2004 🟨 Feb 2004

Tunnels and security

- Packet injection is bad for security
- Z can source arbitrary IPv6 packets from B
  - More effective than IPv6 spoofing
    - Bypasses IPv6 filtering
    - Z can use its real IPv6 source address and receive replies
  - More effective than source routing
    - When packet arrives at B, Hop Limit is untouched
      - ND packets can be spoofed
    - Can't be turned off on routers



A	B	Z	X	Payload

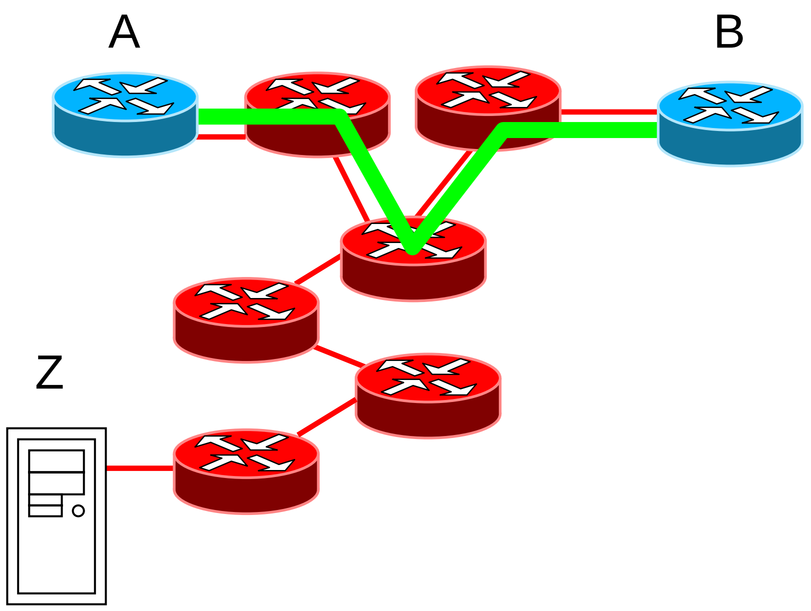
Encapsulated IPv6 packet

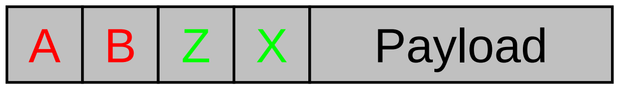
A = IPv4 address of A

A = IPv6 address of A

Tunnels and security (2)

- Packet injection allows Z to:
  - Bypass firewalls / ingress filters
  - Spoof ND packets
    - Redirect, L2 address spoofing, …
    - Not tested, but possibly dangerous
  - …
- What can be done?
  - IPv4 filtering helps
    - But not for interdomain tunnels
  - Don’t trust tunnels and keep them at the edge
  - Use GRE / keyed GRE tunnels





Encapsulated IPv6 packet

A = IPv4 address of A

A = IPv6 address of A

Conclusions

- Tunnel detection
  - Native / tunneled path detection is easy
  - Finding more than one tunnel in a path is harder
  - Finding the endpoints is very difficult
    - Problem: incomplete / inaccurate DNS information
- 6bone database
  - 50% of tunnels nonexistent, 25% working
- IPv6 largely relies on tunnels
  - Seen from 6bone, 8% of paths native
  - Even “native” networks see less than 40% native
  - The situation is slowly improving

Questions?