IPv6 tunnel discovery

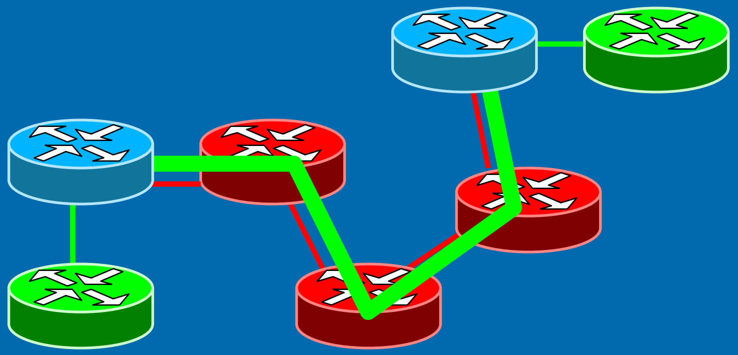
Lorenzo Colitti

Roma Tre University

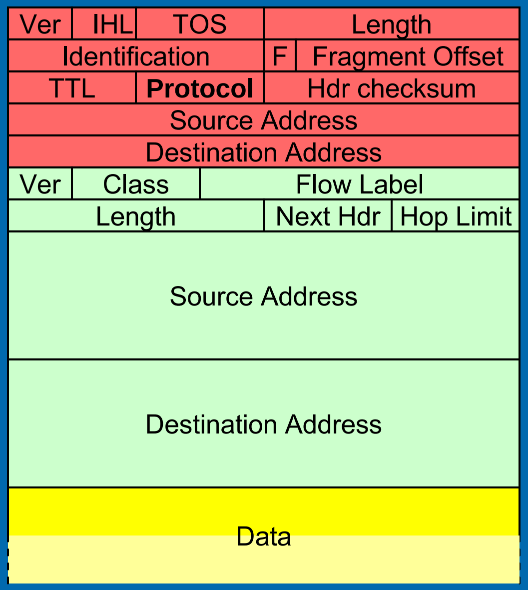
RIPE NCC

IPv6-in-IPv4 tunnels

- Point-to-point link between two routers
- IPv6 uses IPv4 as its “link layer”
- IPv6 packets are encapsulated in raw IPv4 packets with Protocol set to 41



IPv4 Header



Ver | IHL | TOS | Length
Identification | F | Fragment Offset
TTL | **Protocol** | Hdr checksum
Source Address
Destination Address
Ver | Class | Flow Label
Length | Next Hdr | Hop Limit
Source Address
Destination Address
Data

IPv6 Packet

🟥 IPv4 link
🟩 IPv6 link
🟩 IPv6-in-IPv4 tunnel

🟥 IPv4 router
🟩 IPv6 router
🟦 Dual stack router

Problems with tunnels

- Easy migration from IPv4, but:
  - Low performance
    - Place heavy load on routers
    - May lead to inefficient routing
  - Difficult to troubleshoot
  - Pose security problems
- To avoid them we must know they’re there
  - Transparent to IPv6, “single-hop”
  - What can we do?
  - (What we can’t do: DNS)

Objective

- Find all the v6-in-v4 tunnels in the Internet
  - How?
- Start by finding tunnels on one path (A->B)
  - If this can be done, solve the problem (in principle) by iterating over all possible paths
  - For each path, find out:
    1. If the path is native or tunneled (easy)
    2. Which hops are tunnels (more difficult)
    3. The IPv4 endpoints of the tunnels (very difficult)

Approach

- Basic tools:
  - Path MTU discovery
  - Query Internet data sources (DNS, AS, registries)
  - Tunnel endpoint spoofing
  - Hop Limit (= TTL) manipulation
  - Routing Header (= source routing)
- Combine these to form rules
- Each rule has one or more of the following objectives:
  - Infer the existence of tunnels
  - Confirm their existence
  - Collect information about tunnel endpoints
  - Perform “third-party exploration”

Path MTU discovery

- MTU = largest packet a link can carry
- Path MTU discovery reports MTU decreases along path
- Allows us to find first tunnel in a path
  - MTU of tunnel usually lower than native links
  - Certain MTU values typical of tunnels
- Enough if we only want to know if path is native or not

MTU=1500

MTU=1480
⇨Tunnel?

MTU=1500

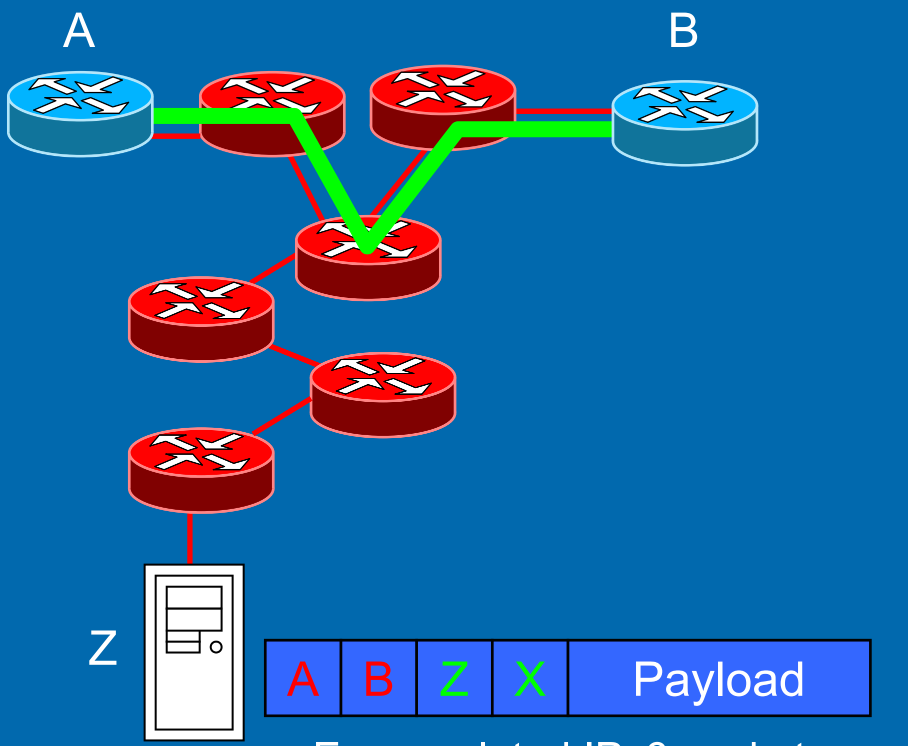
```
colitti@giga:~$ findmtu www.6net.org
1460 (2001:610:16:2000::2 1480, 2001:610:16:2000::2 1460, 2001:610:148:dead:210:18ff:fe02:e38 reached)
colitti@giga:~$ findmtu orange.kame.net
1500 (2001:200:0:8002:203:47ff:fea5:3085 reached)
colitti@giga:~$ 
```

Useful data sources

- DNS
  - Names are protocol-independent
  - Look up v4 address given v6 address (or vice versa)
  - If we suspect a tunnel, DNS can tell us the endpoint
- AS lookups
  - Many tunnels are between two different ASs
  - IPv4 address of an interface may be in different AS from IPv6 address of the same interface
- Registry queries
  - The 6bone registry contains information about tunnels

Endpoint spoofing

- Tunnels provide no authentication mechanism
- Any host Z that knows the IPv4 endpoints can “inject” packets into the tunnel
- We can send IPv6 packets as if connected to B
  - Similar to source routing, but Hop Limit is untouched
- Allows us to:
  - Confirm the presence of a tunnel (send a packet to ourselves)
  - Find more tunnels from B with MTU discovery (fragment encapsulating IPv4 packets)
  - Find IPv6 addresses of A and B using Hop Limit manipulation and source routing



Encapsulated IPv6 packet

A = IPv4 address of A

A = IPv6 address of A

“Tunneltrace”

- A possible algorithm to find tunnels in a path:
  - Look for one tunnel at a time
  - Use Path MTU discovery to find next tunnel
  - Use DNS to discover tunnel endpoints
  - Use endpoint spoofing to confirm tunnel
  - Use endpoint spoofing to carry on discovery process from last tunnel
  - Repeat until end of path reached
- Unfortunately, DNS does not provide enough information to make this feasible
  - Information in DNS is incomplete, inaccurate, or old
- Is there another way?

What else?

- What else can we do?
  - Endpoint spoofing tells us if a tunnel is working or not
    - Given two addresses see if there's a tunnel between them
    - Use this to verify the quality of data in the 6bone registry
      - How many tunnels in the registry are actually working?
  - Path MTU discovery tells us if a path is native or not
    - What percentage of the Internet is native?
    - For example, measure MTU to every prefix in the BGP table
      - This gives us one view of the Internet
      - Endpoint spoofing lets us to do this from any tunnel we know
      - For example, from all the tunnels in the 6bone database

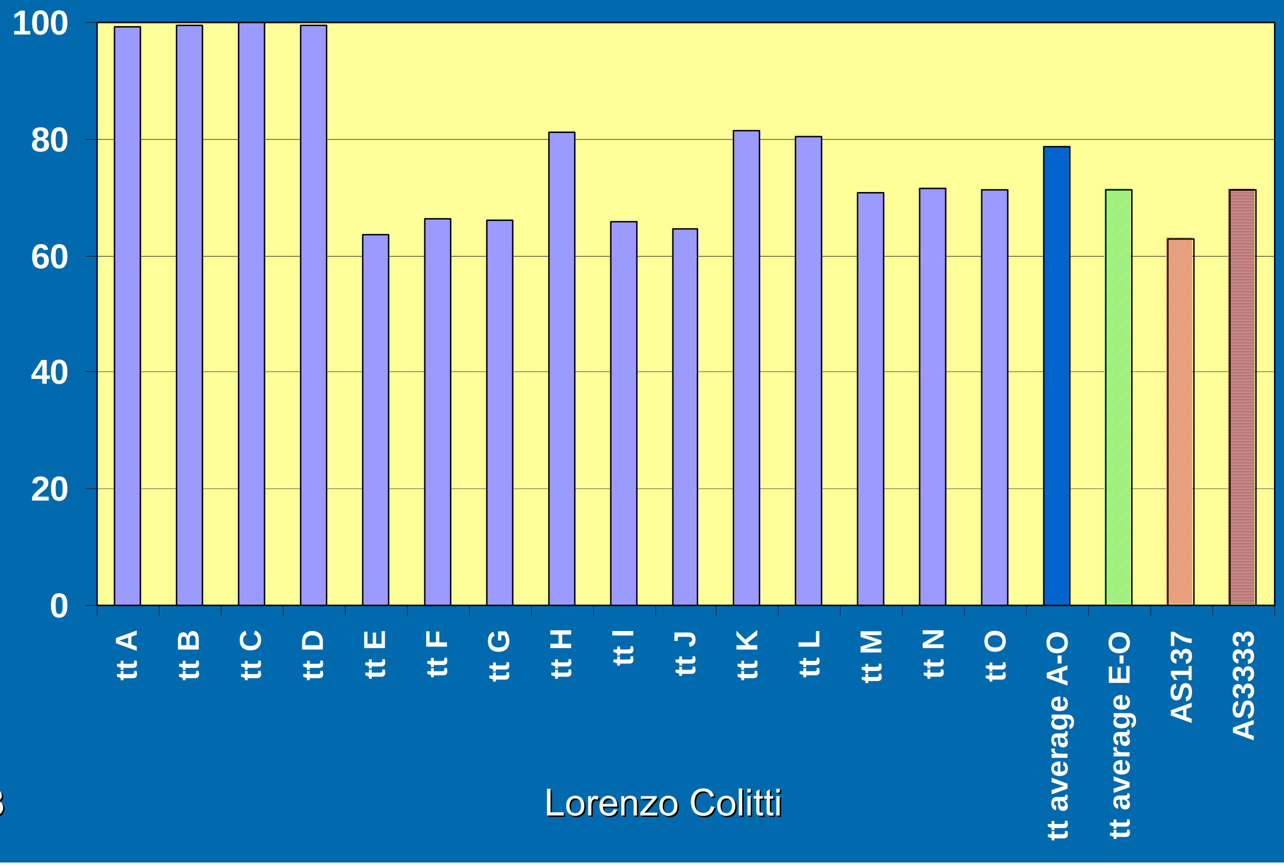
How many tunnels?

- The 6bone registry contains >1000 “working” tunnels (~25%)
- The rest nonexistent (~50%), down or filtered
- We measured the MTU from each endpoint to all prefixes in the global IPv6 BGP table
- Result: native paths only ~ 8% of total

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

How native is “native”?

Percentage of BGP prefixes reached through tunnel(s) from GARR and RIPE NCC / testboxes. Even the “best” are ≥ 62%



Tunnel detection in TTM

- The TTM service will soon use tunnel detection to qualify IPv6 data
- Every testbox measures MTU to every other testbox and results are displayed in a matrix
- Historical data will provide information on routing changes
- Most testboxes are in native IPv6 networks

http://www.ripe.net/ripencc/mem-services/ttm/Plots/pmtu/tunneldiscovery.cgi

		Destination Testbox																
		tt01	tt107	tt13	tt25	tt35	tt42	tt52	tt55	tt56	tt72	tt73	tt76	tt77	tt85	tt86	tt94	tt97	tt98
Source Testbox	tt01		1500	1500	1280	1500	1500	1500	1500	1280	1500	1500	1500	1500	1500	1500	1500	1500	1500
	tt107	1500		1500	1280	1500	1280	1500	1500	1280	1500	1500	1500	1476	1476	1476	1500	1500	1500
	tt13	1500	1500		1280	1500	1500	1500	1500	1280	1500	1500	1500	1500	1500	1500	1500	1500	1500
	tt25	1480	1476	1480		1476	1476	1476	1476	1280	1476	1476	1476	1476	1476	1476	1480	1480	1480
	tt35	1500	1500	1500	1476		1480	1500	1500	1280	1500	1500	1500	1476	1476	1476	1500	1500	1500
	tt42	1500	1500	1500	1476	1500		1500	1500	1280	1500	1500	1500	1500	1500	1500	1500	1500	1500
	tt52	1500	1500	1500	1280	1500	1480		1500	1280	1500	1500	1500	1476	1476	1476	1500	1500	1500
	tt55	1500	1500	1500	1280	1500	1480	1500		1280	1500	1500	1500	1476	1476	1476	1500	1500	1500
	tt56	1476	1476	1476	1280	1476	1476	1476	1280		1280	1476	1476	1476	1476	1476	1476	1476	1476
	tt72	1500	1500	1500	1280	1500	1480	1500	1500	1280		1500	1500	1476	1476	1476	1500	1500	1500
	tt73	1480	1500	1500	1476	1500	1500	1500	1500	1280	1500		1500	1500	1500	1500	1480	1480	1480
	tt76	1480	1480	1480	1480	1480	1480	1480	1480	1476	1480	1480		1480	1480	1480	1480	1480	1480
	tt77	1500	1476	1500	1476	1476	1500	1500	1476	1280	1476	1476	1476		1500	1500	1500	1500	1500
	tt85	1500	1476	1500	1480	1476	1500	1500	1476	1280	1476	1500	1500	1476		1500	1500	1500	1500
	tt86	1500	1476	1500	1476	1476	1500	1500	1476	1280	1476	1500	1500	1476	1500		1500	1500	1500
	tt94	1500	1500	1500	1280	1500	1500	1500	1500	1280	1500	1500	1500	1500	1500	1500		1500	1500
	tt97	1500	1500	1500	1280	1500	1500	1500	1500	1280	1500	1500	1500	1500	1500	1500	1500		1500
	tt98	1500	1500	1500	1280	1500	1500	1500	1500	1280	1500	1500	1500	1500	1500	1500	1500	1500	

http://www.ripe.net/ripencc/mem-services/ttm/Plots/pmtu/tracepath.cgi?sourceBox=tt56&destBox=tt86&pmtu=1068419744 tt56.ripe.net tt86.ripe.net 1476 (2001:7d0...

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 and/or inaccurate DNS information
- 6bone database
  - 50% of tunnels nonexistent
  - 25% working
- IPv6 largely relies on tunnels
  - In total, 8% of paths native
  - Even “native” networks are not better than 40% native

TR available at:
http://web.dia.uniroma3.it/ricerca/rapporti/schedaRapporto.php?id=82