IPv6 Alias Resolution via Induced Fragmentation

Billy Brinkmeyer, Robert Beverly, Matthew Luckie*, Justin Rohrer

Naval Postgraduate School
*CAIDA
{wdbrinkm,rbeverly,jprohrer}@nps.edu
mjl@caida.org

February 6-8, 2013

ISMA 2013 AIMS-5 - Workshop on Active Internet Measurements

Problem Overview

The Problem:

- What is the *topology* of the IPv6 Internet?
- We tackle initial work on the “alias resolution” problem for IPv6 to infer *router-level* topologies.

- Given two IPv6 addresses, determine whether they are assigned to *different* interfaces on the *same* physical router.

Prior Work (IPv6)

Prior Work (IPv6)

- All previous work relies on IPv6 source-routing (questionable long-term?).
- Waddington, et al. (2003): Atlas. Source-routed, TTL-limited UDP probe to y via x. Assuming v6 routing header processed first and (x, y) are aliases $\rightarrow$ receive "hop limit exceeded" and "port unreachable."
- Qian, et al. (2010): Route Positional Method. Send TTL-limited UDP probe to self via x and y. If aliases $\rightarrow$ receive TTL expiration from x.
- Qian, et al. (2010): Same idea, but using invalid bit sequence in IPv6 option header.
- The Hacker's Choice (THC) v6 attack toolkit: reduce IPv6 MTU.

Prior Work (IPv6)

Prior Work (IPv6)

- All previous work relies on IPv6 source-routing (questionable long-term?).
- Waddington, et al. (2003): Atlas. Source-routed, TTL-limited UDP probe to y via x. Assuming v6 routing header processed first and (x, y) are aliases $\rightarrow$ receive "hop limit exceeded" and "port unreachable."
- Qian, et al. (2010): Route Positional Method. Send TTL-limited UDP probe to self via x and y. If aliases $\rightarrow$ receive TTL expiration from x.
- Qian, et al. (2010): Same idea, but using invalid bit sequence in IPv6 option header.
- The Hacker's Choice (THC) v6 attack toolkit: reduce IPv6 MTU.

IPv6 Fragmentation

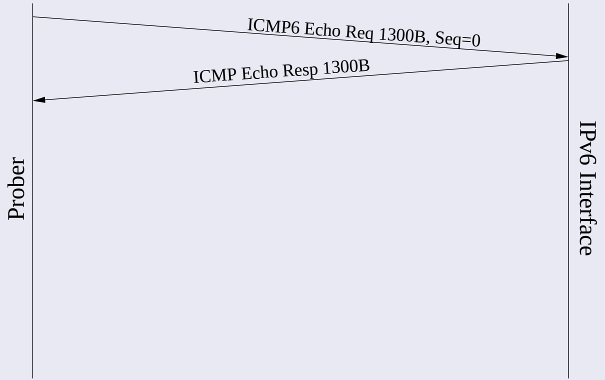
Eliciting Fragmented Responses

- We take inspiration from prior IPv4 IPID work
- But... no in-network fragmentation in IPv6 (push all work to end-hosts)
- If a router's next hop interface's MTU is less than the size of a packet, it sends an ICMP6 "packet too big" message to the source [RFC2460]
- End-host maintains destination cache state of per-destination maximum MTU
- End-hosts can fragment packets using an IPv6 fragmentation header

Too-Big Trick

Too-Big Trick

- “*IPv6 Alias Resolution via Induced Fragmentation*” (to appear: PAM 2013)

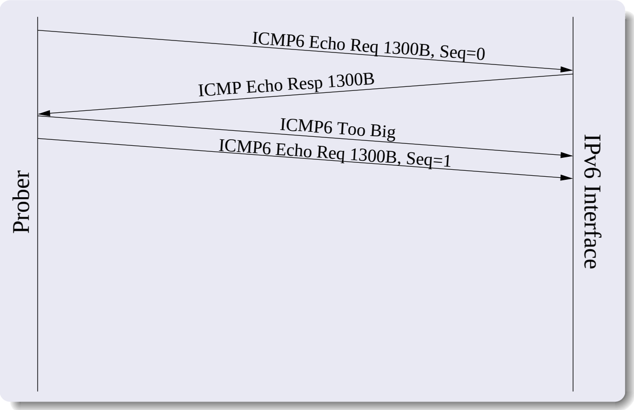


Send a 1300 byte ICMP6 echo request to router interface

Too-Big Trick

Too-Big Trick

- Induce a remote router to originate fragmented packets

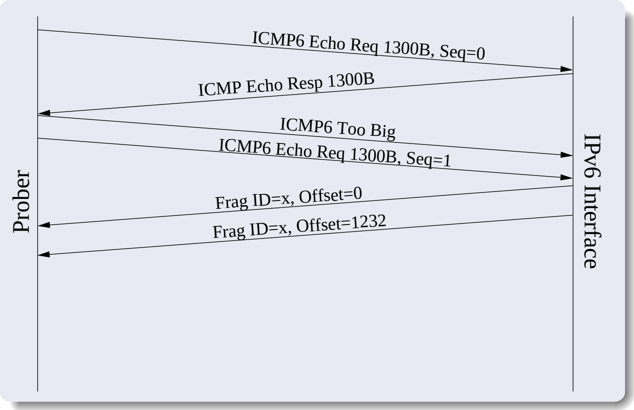


Ignore response. Send ICMP6 packet-too-big message. Send new ICMP6 echo request.

Too-Big Trick

Too-Big Trick

- Induce a remote router to originate fragmented packets

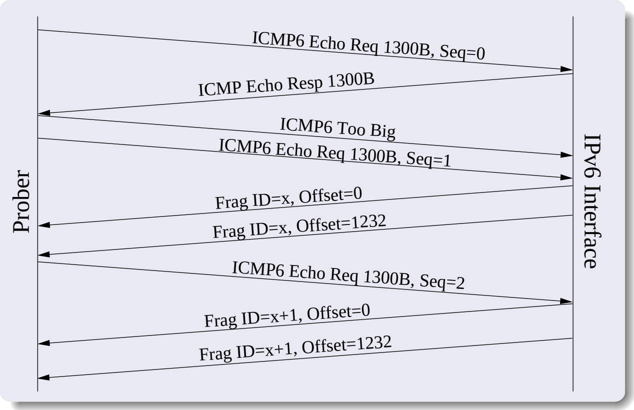


Router replies with fragmented ICMP6 echo response.

Too-Big Trick

Too-Big Trick

- Induce a remote router to originate fragmented packets



Prober can elicit new fragment identifiers with each ICMP6 echo request.

How Effective is TBT on the Internet?

Efficacy of TBT

- Determine *how many* live IPv6 interfaces respond to TBT
- Determine *in what way* they respond

Methodology:

- Single vantage point
- TBT probe 49,000 interfaces:
  - 23,892 distinct IPv6 interfaces from CDN traceroutes (May, 2012)
  - 25,174 distinct IPv6 interfaces from CAIDA (August, 2012)
- Includes IPv6 router interfaces in 2,617 autonomous systems
- Check for liveness
- Elicit 10 fragment IDs (20 total fragments)

TBT Response Characteristics

TBT Response Characteristics

	CDN		CAIDA	
ICMP6 responsive	18486/23892	77.4%	18959/25174	75.3%
Post-TBT unresp.	235/18486	1.3%	66/18959	0.4%
Post-TBT nofrags	5519/18486	29.9%	5800/18959	30.6%

- Of interfaces responding to “normal” ICMP6 echo request:
  - $\approx$ 30% do not send fragments after TBT
  - $\approx$ 1% become unresponsive!

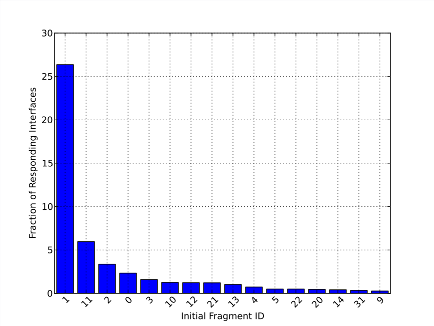
TBT Response Characteristics

TBT Response Characteristics

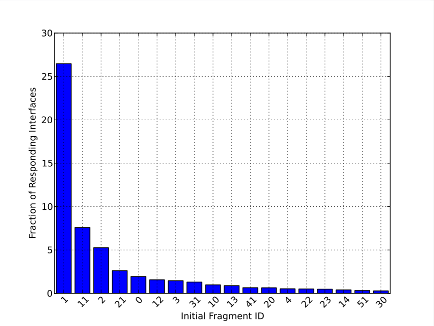
	CDN		CAIDA	
TBT responsive	12732/18486	68.9%	13093/18959	69.1%
TBT sequential	8288/12732	65.1%	9183/13093	70.1%
TBT random	4320/12732	33.9%	3789/13093	28.9%

- Thus, $\approx$ 70% return fragment identifiers after TBT
- Of those:
  - $65 - 70\%$ return *sequential IDs*!
  - (Unfortunately, *not* same as IPv4 ID)
  - Remaining $\approx$ 30% use random IDs (confirmed as Juniper)

Initial Fragment Identifiers



CDN



CAIDA

- $\approx$ 25% of interfaces responded with fragment ID=1 after first probe
- These routers sent *no* fragmented traffic prior to our probe!
- Observe: modes at multiples of 10. Naturally discovering aliases!

IPv6 Alias Resolution Algorithm

IPv6 Alias Resolution using TBT:

- IPv6 control plane traffic does not "spin" counter (unlike IPv4)
- Can reasonably expect IPv6 identifiers to have no natural velocity over probing interval
- IPv6 fragment identifiers are 32-bit (unlike IPv4)

Caveats

- Many routers will have low fragment identifiers
- Fragment counter may be the same for many routers
- Intuition: cause counters of non-aliases to diverge
- Probe candidate pair (A, B) at different rates

IPv6 Internet Alias Resolution

Controlled Environment

- Used GNS3 to build a virtualized 26-node Cisco network running IOS 12.4(20)T
- Found that Cisco uses sequential IPv6 fragment IDs
- Validated TBT and algorithm: 100% accuracy (f-score = 1.0) in finding 92/92 aliases (1584/1584 non-aliases)

IPv6 Internet Alias Resolution

- Worked with a commercial service provider to get ground-truth on 8 physical routers in production
- Each of 8 routers has 2-21 IPv6 interfaces
- Using TBT, correctly identified 808/808 true aliases, with no false positives

Large-Scale IPv6 Alias Resolution

Large-Scale IPv6 Alias Resolution

- PAM paper only demonstrates technique and feasibility
- Algorithm in PAM paper is inefficient: $O(N^2)$.
- Instead, NPS/CAIDA have begun investigating a new algorithm (ask us for details).

Large-Scale IPv6 Alias Resolution

Initial Controlled Large-Scale Testing

- Again, used GNS3: 26 virtual routers

	naïve TBT	LS-TBT	Savings
Pings	8968	222	98%
Time	36:33	4:24	$\approx$ 1/10 time
Aliases	54/54	54/54	-

- Promising start
- Work proceeding on Internet-wide probing

Summary

Summary:

- New fingerprinting-based IPv6 alias resolution technique
- Internet-wide probing of $\approx 49,000$ live IPv6 interfaces, 70% of which respond to our test
- Validation of technique on subset of production IPv6 network
- ScaPy implementation: `http://www.cmand.org/tbt`
- (Now implemented in scamper; ask mjl)
- Eventual plan: release v6 aliases as part of CAIDA ITDK

Thanks! From audience:

- Better understanding of our TBT-induced failures?
- Any other v6 networks for ground-truth evaluation?
- Thoughts on v4/v6 associations for routers?

IPv6 Alias Resolution Algorithm

```
send(A, TooBig)
send(B, TooBig)
for i in range(5) do
    ID[0] ← echo(A)
    ID[1] ← echo(B)
    if (ID[0]+1) ≠ ID[1] then
        return False
    ID[2] ← echo(A)
    if (ID[1]+1) ≠ ID[2] then
        return False
return True
```

Large-Scale IPv6 Alias Resolution

Algorithm Intuition by Example

- Let *A* be an IPv6 router with 3 interfaces, *B* 2 interfaces, *C* 1 interface, *D* 2 interfaces.
- Assume initial fragment ID state:

A	B	C	D
1	1	1	9

Large-Scale IPv6 Alias Resolution

- Spin all interfaces, get back ID^1:

A1	A2	A3	B1	B2	C1	D1	D2
2	3	4	2	3	2	10	11

- Spin all again. Get back ID^2:

A1	A2	A3	B1	B2	C1	D1	D2
5	6	7	4	5	3	12	13

Observe:

- Any interface where $ID^1 + 1 = ID^2$: no aliases of that interface (because ID^2 would have to be $> ID^1 + 1$, eliminate. Here, eliminate $C1$.
- More generally, # aliases of an interface = $ID^2 - ID^1$.
- Therefore: $A1$, $A2$, $A3$ are *possible* aliases

Large-Scale IPv6 Alias Resolution

- Spin all interfaces, get back ID^1:

A1	A2	A3	B1	B2	C1	D1	D2
2	3	4	2	3	2	10	11

- Spin all again. Get back ID^2:

A1	A2	A3	B1	B2	C1	D1	D2
5	6	7	4	5	3	12	13

Observe:

- Other constraints given population: $D1$, $D2$ must be aliases (no other ID=13 exists).
- Further, $A1$, $B2$ *cannot* be aliases.
- Disambiguate remaining candidates using TBT PAM work.

Work beyond PAM Paper

End-Host Responsiveness

- Technique can also be applied to end-hosts (which may have multiple v6 interfaces)

Operating System	Initial Fragment ID	Subsequent Frag IDs
Ubuntu	Random	Sequential
Fedora	Random	Sequential
FreeBSD	Random	Random
OpenSUSE	Random	Sequential
Windows XP	1	Sequential
Windows 2003 Server	1	Sequential
Windows 7	0	2,4,6,8,...