Inferring Internet Server IPv4 and IPv6 Address Relationships

Robert Beverly, Arthur Berger*, Nicholas Weaver†, Larry Campbell*

Naval Postgraduate School
*Akamai
†ICSI/UCSD
rbeverly@nps.edu, awberger@mit.edu

February 7, 2013

CAIDA Active Internet Measurement 2013

Sibling Resolution

New Problem We Term "Sibling Resolution:"

Given a candidate (*IPv*4, *IPv*6) address pair, determine if these addresses are assigned to the same cluster, device, or interface.

- Lots of prior work on passive sibling associations: e.g. web-bugs, javascript, flash, etc.
- Prior work focuses on clients (adoption, performance)
- This work:
  - *Targeted, active test:* on-demand for any given pair
  - *Infrastructure:* finding server siblings
- Eventual goal: router siblings (not there yet)

Motivation

Why?

- Adoption (non-adoption):
  - IPv4 and IPv6 expected to co-exist (for a long while?) $\rightarrow$ dual-stacked devices
  - Track IPv6 evolution
- Security:
  - IPv6 is largely unsecured!
  - Inter-dependence of IPv6 on IPv4 (and vice-versa)
  - e.g. attack on IPv6 resource affecting IPv4 service
  - Correlating geolocation, reputation, etc with IPv4 host counterpart.
- Performance:
  - Getting measurements of IPv4 vs. IPv6 performance correct: isolate path vs. host performance
- Operationally deployed today in Akamai, informing Edgescape geolocation.

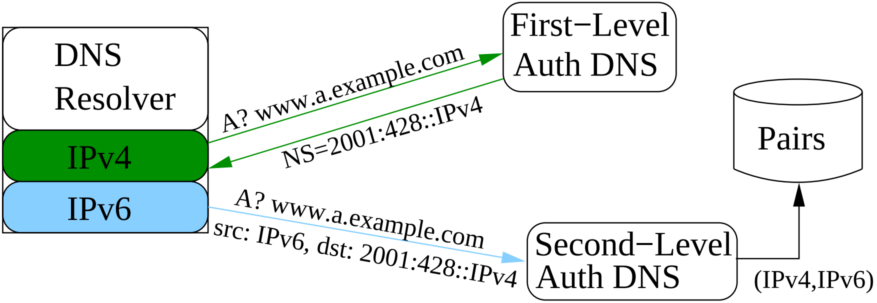
Techniques

3 Techniques:

1. **(Passive)** Induce DNS resolvers to use both v4 and v6 during natural resolution of Akamai resources (deployed, large set of measurements).
2. **(Active)** Force DNS to use a *chain* of v4 and v6 addresses to perform resolution. Allows us to validate (a subset) of the passively collected results.
3. **(Active)** Probe potentially in-common TCP stack of a candidate v4, v6 sibling pair to obtain timestamp fingerprint.

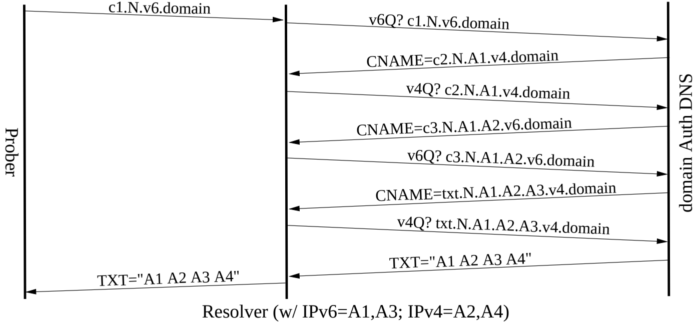
Passive DNS

- Encode IPv4 address of querying resolver into a `AAAA` record returned for the next-level `NS`
- Subsequent query to the IPv6 authority nameserver permits linking v4 and v6 resolver addresses



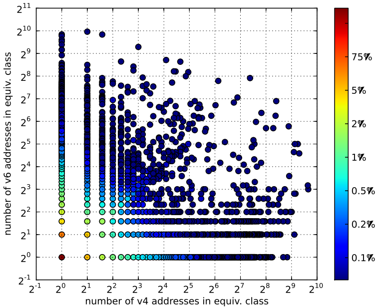
Active DNS

- Custom DNS server as authority for special domain
- Chain of alternating v6, v4 `CNAME` records, only available via v6 or v4, that maintain state within the dynamic name.



DNS Results

- Deployed on Akamai; gathered $\simeq$ 675,000 v4,v6 pairs
- Importance: directing users to content in a CDN relies on properties of DNS resolution. Improves IPv6 geolocation.
- 77% of v4,v6 pairs are 1-1, the rest is messy. Most complexity due to large cluster resolvers (e.g. nominum, google DNS, openDNS, comcast, etc).



Targeted, Active Technique

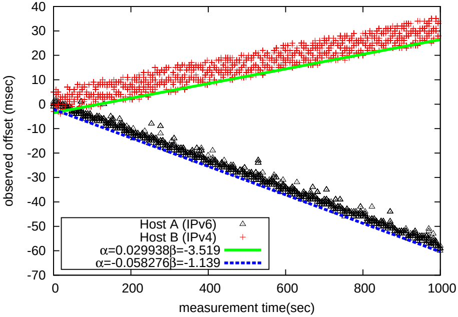
- Intuition: IPv4 and IPv6 share a common transport-layer (TCP) stack
- Leverage prior work on physical device fingerprinting using TCP timestamp clockskew [Kohno 2005]
- TCP timestamp option: “TCP Extensions for High Performance” [RFC1323, May 1992]. Universally supported, enabled by default.
- Note: TS clock $\neq$ system clock
- Note: TS clock frequently unaffected by system clock adjustments (e.g. NTP)
- ***Basic Idea:*** Probe over time. Fingerprint is clock *skew* (and remote clock resolution).

Example

Example

- Gather 4 timestamp series:
  - `www.caida.org` (v4 and v6)
  - `www.ripe.net` (v4 and v6)

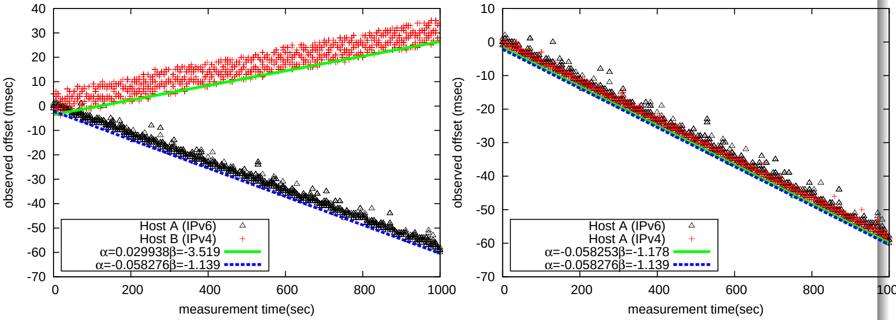
Example



CAIDA IPv6 vs. RIPE IPv4

- Observe different skew slopes (one negative)
- Different timestamp granularity
- $y = 0.029938x$ equates to skew of $\approx$ 1.8ms / minute, or $\approx$ 15 minutes per year.
- False siblings!

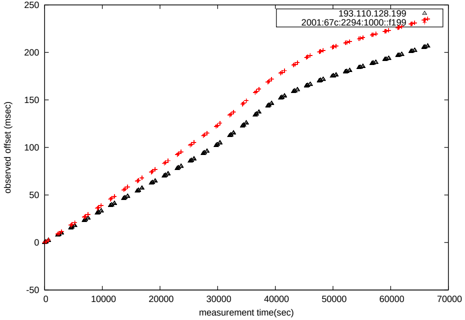
Example



False Siblings True Siblings

- CAIDA IPv4 vs. CAIDA IPv6: identical slopes ($\theta = 0.0098$)
- CAIDA IPv6 vs. RIPE IPv4: different slopes ($\theta = 31.947$)

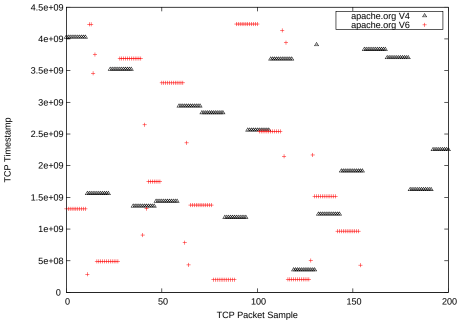
Complications



www.marca.com (#6 on alexa ipv6)

- Not always so distinct of a difference!
- Slope angle difference: $\theta = 2.046$

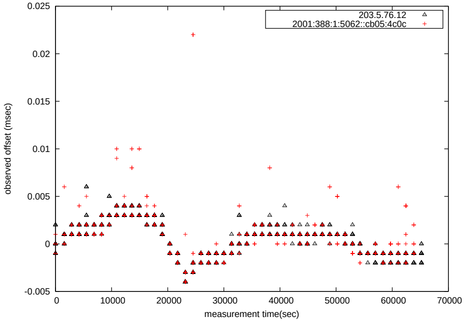
Complications



www.apache.com

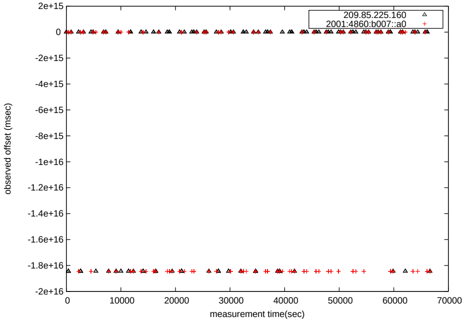
- Raw TCP timestamps
- Deterministically random and monotonic for a single connection
- Random across connections. Looks like noise to us.

Complications



- What's going on here?

Complications



- Also detects load balancing among servers
- But how to deal with it?

Machine Sibling Inference

Machine Sibling Inference Methodology:

- Analyze Alexa top 100,000 websites
- Pull `A` and `AAAA` records
- 1398 ($\approx$ 1.4%) have IPv6 DNS
- Repeatedly fetch root HTML page via IPv4 and IPv6 via deterministic IP address
- Record all packets

Machine Sibling Inference

Alexa 100K Targeted Machine-Sibling Inference

Case	Count
v4 and v6 non-monotonic (possible siblings)	109 (7.8%)
v4 or v6 non-monotonic (non-siblings)	140 (10.0%)
v4 and v6 no timestamps (possible siblings)	94 (6.7%)
v4 or v6 no timestamps (non-sibling)	101 (7.2%)

- Our technique fails when timestamps are not monotonic across TCP flows (e.g. load-balancer or BSD OS)
- Or, when timestamps are not supported (e.g. middlebox)
- Note, can disambiguate non-siblings

Machine Sibling Inference

Alexa 100K Targeted Machine-Sibling Inference

Case	Count
v4 and v6 non-monotonic (possible siblings)	109 (7.8%)
v4 or v6 non-monotonic (non-siblings)	140 (10.0%)
v4 and v6 no timestamps (possible siblings)	94 (6.7%)
v4 or v6 no timestamps (non-sibling)	101 (7.2%)
Skew-based siblings	839 (60.0%)
Skew-based non-siblings	115 (8.3%)
Total	1398 (100%)

- 25.5% (356) non-siblings
- 57% of skew-based non-siblings are in *same* AS
- 12.6% of skew-based siblings are in *different* ASes

Feedback

Thanks!

- **Viz**: Awesome scatter plot!
- **Data-Sharing**: None so far (Akamai data off-limits, web-probing can be released)
- **Feedback**:
  - Do you believe our motivation story!?!?
  - Operational experience with large DNS resolvers?
  - Thoughts on router v4,v6 sibling resolution?