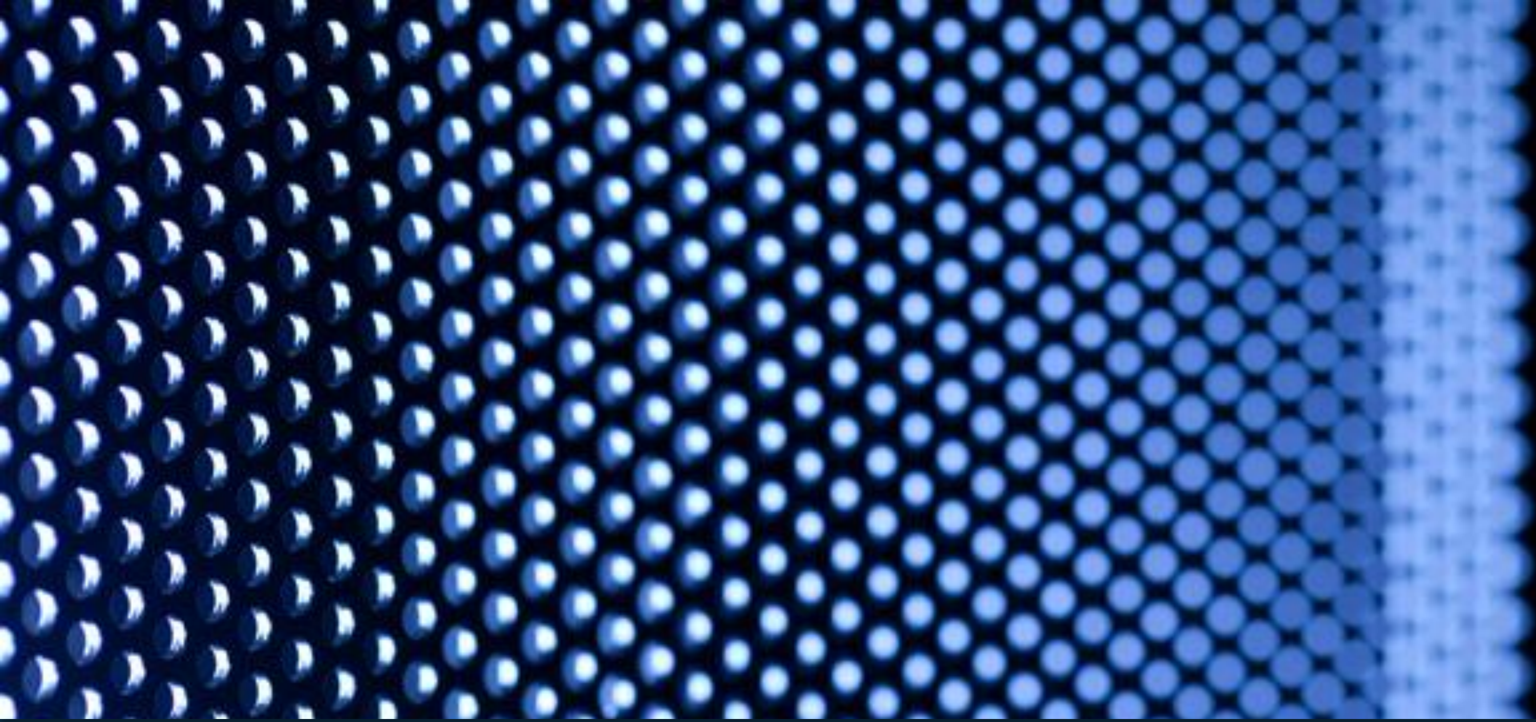


Query name minimization and authoritative DNS server behavior

Shumon Huque, Verisign Labs

DNS-OARC 2015 Spring Workshop, Amsterdam, Netherlands

May 9th 2015

Title: Query-name minimization and authoritative DNS server behavior
Venue: DNS-OARC, May 9th 2015, Amsterdam, NL
Speaker: Shumon Huque

Abstract:

This talk will provide an overview of DNS query name minimization algorithms (an approach to improving DNS privacy), and discuss the behavior of some authoritative DNS servers in the presence of a resolver that performs query name minimization. I'll provide a survey of results of the Alexa top 1000 domains, and discuss some observed defective behavior of several CDNs and DNS hosting providers that will need to be addressed before these new resolution algorithms can be used widely.

Introduction

- A brief examination of the behavior of some authoritative Domain Name System (DNS) servers in the presence of a resolver that performs *query name minimization*.

Query name minimization

- An approach to iterative DNS resolution that aims to provide enhanced privacy preserving properties to the resolver.
- Exposes only the minimum portion of the query name needed to authoritative servers as it traverses the DNS delegation hierarchy to the target zone.
- Gradually prepends labels from the full query name as it follows referrals and descends zones.

- Internet Engineering Task Force (IETF) protocol specification in progress:
  - https://tools.ietf.org/html/draft-ietf-dnsop-qname-minimisation

Query name minimization

- Note: requires no changes to the DNS wire protocol.
- Resolvers can unilaterally decide to use this resolution method, without any co-operation from other parties.

- But does it actually work in practice?

Minimization algorithm (simplest)

- For **www.example.com.**, **IN**, **A:**
  - Send: **com. IN, A** to the **root** servers
    - Obtain referral to the com. servers
  - Send: **example.com. IN, A** to the **com** servers
    - Obtain referral to the example.com servers
  - Send: **www.example.com. IN, A** to the **example.com** servers
    - (Probably) obtain the full answer here

  - If any intermediate qname results in NXDOMAIN, stop and return NXDOMAIN.
  - If any intermediate qname results in an answer (empty or non-empty) rather than a referral, prepend next label and re-query.

Minimization algorithm (variants)

- Use obfuscated (nonce) labels.

- Hide the original query-type too.
  - Send NS record queries (instead of original query type).
  - Lookup address records for NS record targets.
  - Follow referrals, descend zones etc.
  - At the target zone, issue the full query with the original query type.

  - Will this algorithm will be harder to deploy because of middleboxes that might trap NS queries as unexpected, or broken authority servers that don't respond correctly to explicit NS queries?
  - Query types reveal additional info about intent – may need privacy

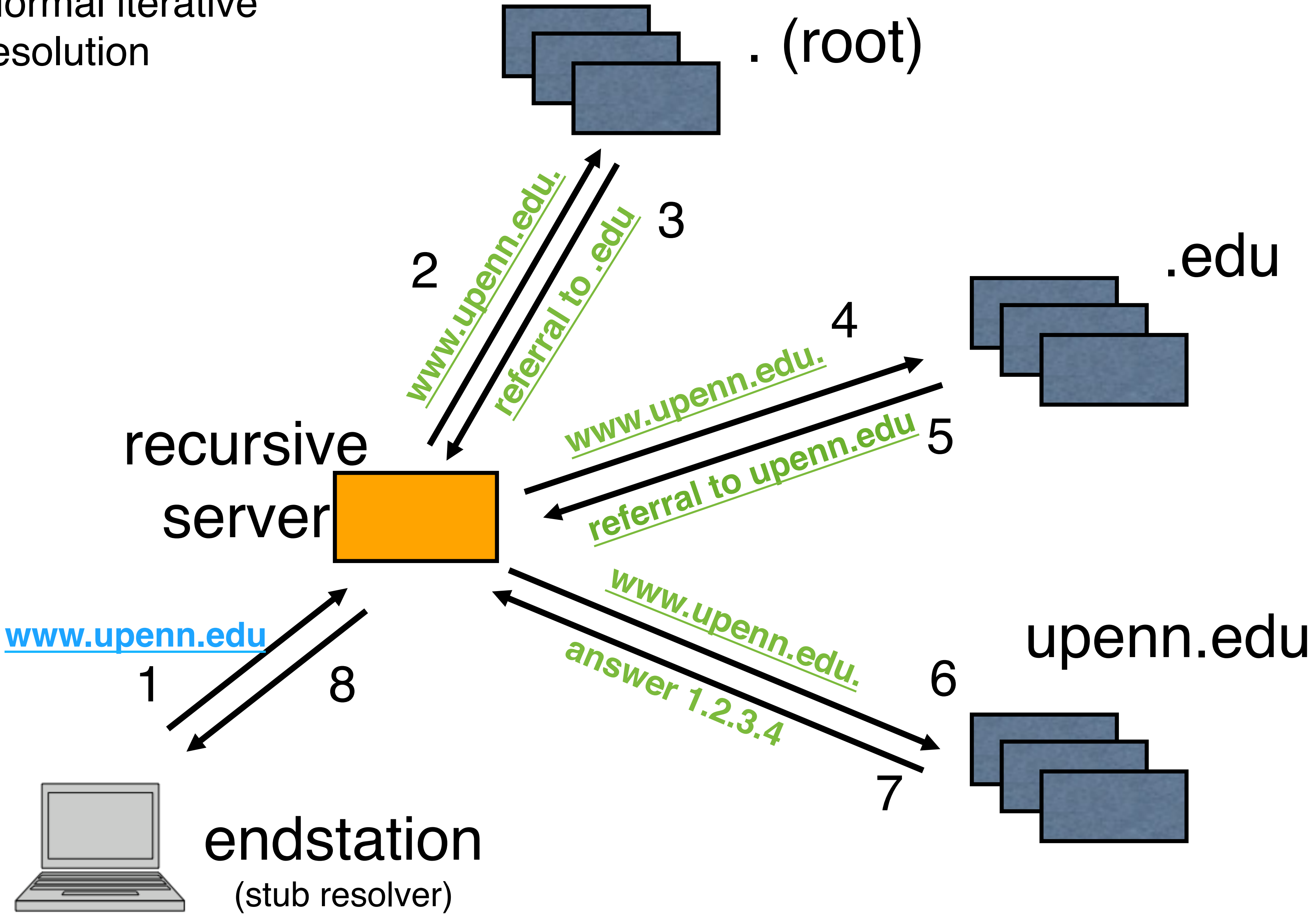
Minimization algorithm (variants)

- Hybrid strategies.
- Aggressive (always minimize) vs. lazy (minimize after learning zone cuts) modes.
- Treat some levels specially, e.g. root and Top-level domains (TLDs) – always send minimized queries to them, and perhaps full queries further down the hierarchy.

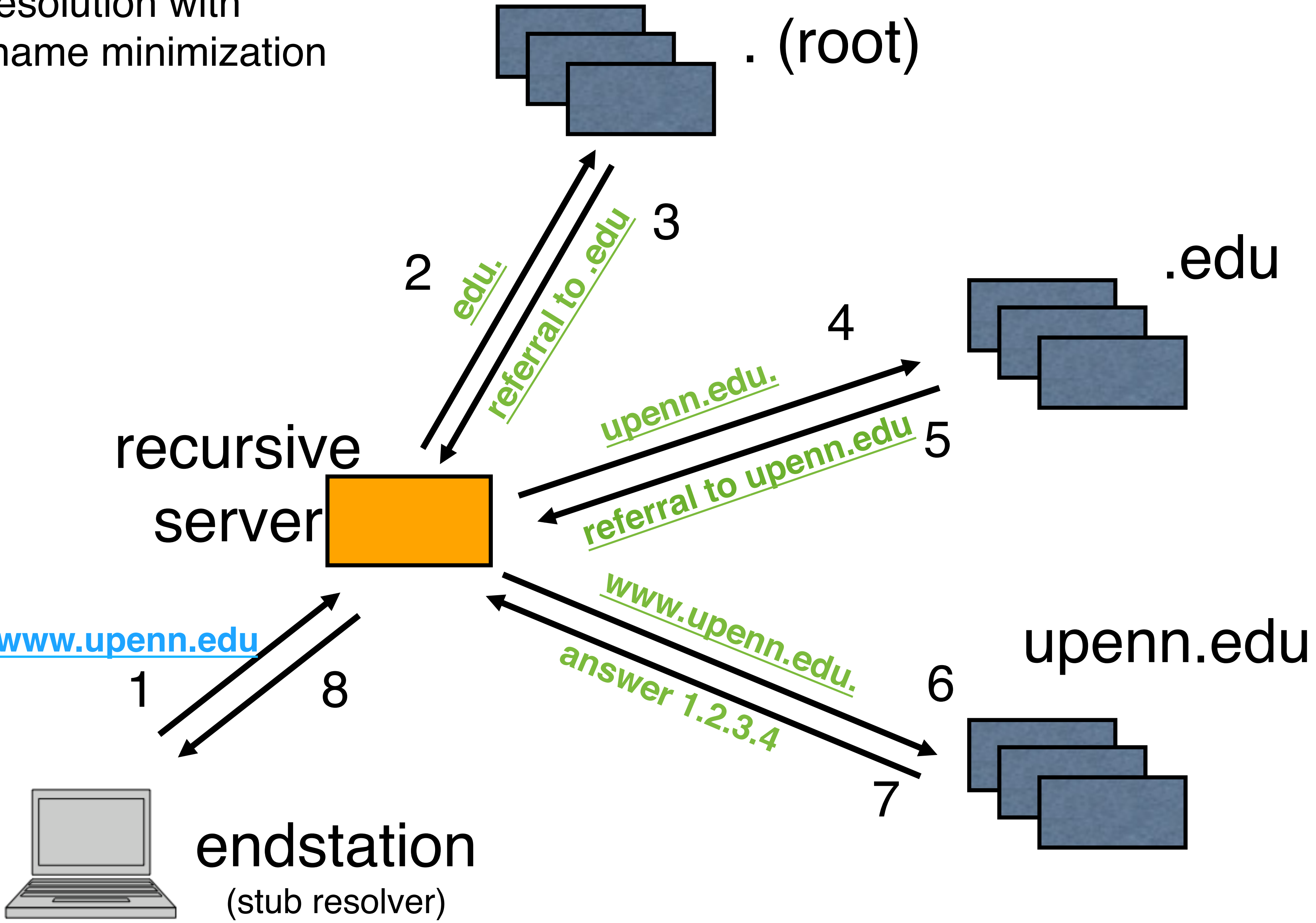
Testing the simple algorithm

- I implemented the simplest algorithm in a standalone iterative resolver.
- Tested it out on a set of selected domain names.
- Also tested the Alexa top 1000 domains to see if I could successfully resolve their website names.

Normal iterative resolution



Resolution with qname minimization



www.amazon.com

```
$ resolve.py www.amazon.com. A

>> Query: www.amazon.com. A IN at zone .
>>        [Got Referral to zone: com.]

>> Query: www.amazon.com. A IN at zone com.
>>        [Got Referral to zone: amazon.com.]

>> Query: www.amazon.com. A IN at zone amazon.com.
>>        [Got Referral to zone: www.amazon.com.]

>> Query: www.amazon.com. A IN at zone www.amazon.com.
www.amazon.com. 60 IN A 176.32.98.166
```

www.amazon.com with minimization

```
$ resolve.py -m www.amazon.com. A

>> Query: com. A IN at zone .
>>         [Got Referral to zone: com.]

>> Query: amazon.com. A IN at zone com.
>>         [Got Referral to zone: amazon.com.]

>> Query: www.amazon.com. A IN at zone amazon.com.
>>         [Got Referral to zone: www.amazon.com.]

>> Query: www.amazon.com. A IN at zone www.amazon.com.
www.amazon.com. 60 IN A 176.32.98.166
```

Examples that don’t work

- Some popular Content Delivery Networks (CDNs) seem to have trouble with minimized query names.
- Many of them work by aliasing a customer website name to a name in a zone they operate.
  - The name is constructed by prepending the customer site name in some fashion to their zone name. And constructing dynamic querier specific answers in the responses.
  - In many cases, intermediate query names in the alias target (usually empty non-terminals) are not handled correctly (not expecting to receive queries for those names?)

www.upenn.edu

Aliased to Akamai CDN

```
$ resolve.py www.upenn.edu. A
>> Query: www.upenn.edu. A IN at zone .
>>          [Got Referral to zone: edu.]
>> Query: www.upenn.edu. A IN at zone edu.
>>          [Got Referral to zone: upenn.edu.]
>> Query: www.upenn.edu. A IN at zone upenn.edu.
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.
>> Query: www.upenn.edu-dscg.edgesuite.net. A IN at zone .
>>          [Got Referral to zone: net.]
>> Query: www.upenn.edu-dscg.edgesuite.net. A IN at zone net.
>>          [Got Referral to zone: edgesuite.net.]
>> Query: www.upenn.edu-dscg.edgesuite.net. A IN at zone edgesuite.net.
www.upenn.edu-dscg.edgesuite.net. 21600 IN CNAME a1165.dscg.akamai.net.
>> Query: a1165.dscg.akamai.net. A IN at zone net.
>>          [Got Referral to zone: akamai.net.]
>> Query: a1165.dscg.akamai.net. A IN at zone akamai.net.
>>          [Got Referral to zone: dscg.akamai.net.]
>> Query: a1165.dscg.akamai.net. A IN at zone dscg.akamai.net.
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.
www.upenn.edu-dscg.edgesuite.net. 21600 IN CNAME a1165.dscg.akamai.net.
a1165.dscg.akamai.net. 20 IN A 23.62.7.50
a1165.dscg.akamai.net. 20 IN A 23.62.7.64
```

(Transcript from February 2015)

www.upenn.edu with minimization

```
$ resolve.py -m www.upenn.edu. A
>> Query: edu. A IN at zone .
>>         [Got Referral to zone: edu.]
>> Query: upenn.edu. A IN at zone edu.
>>         [Got Referral to zone: upenn.edu.]
>> Query: www.upenn.edu. A IN at zone upenn.edu.
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.
>> Query: net. A IN at zone .
>>         [Got Referral to zone: net.]
>> Query: edgesuite.net. A IN at zone net.
>>         [Got Referral to zone: edgesuite.net.]
>> Query: edu-dscg.edgesuite.net. A IN at zone edgesuite.net.
ERROR: NXDOMAIN: edu-dscg.edgesuite.net. not found
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.

[Resolution stops here without yielding an answer]
```

NXDOMAIN answer at
empty non-terminal name

(Transcript from February 2015)

www.ietf.org

Aliased to Cloudflare CDN

```
$ resolve.py www.ietf.org A
>> Query: www.ietf.org. A IN at zone .
>>          [Got Referral to zone: org.]
>> Query: www.ietf.org. A IN at zone org.
>>          [Got Referral to zone: ietf.org.]
>> Query: www.ietf.org. A IN at zone ietf.org.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.
>> Query: www.ietf.org.cdn.cloudflare.net. A IN at zone .
>>          [Got Referral to zone: net.]
>> Query: www.ietf.org.cdn.cloudflare.net. A IN at zone net.
>>          [Got Referral to zone: cloudflare.net.]
>> Query: www.ietf.org.cdn.cloudflare.net. A IN at zone cloudflare.net.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.
www.ietf.org.cdn.cloudflare.net. 300 IN A 104.20.0.85
www.ietf.org.cdn.cloudflare.net. 300 IN A 104.20.1.85
```

(Transcript from February 2015)

www.ietf.org with minimization

```
$ resolve.py -m www.ietf.org A
>> Query: org. A IN at zone .
>>        [Got Referral to zone: org.]
>> Query: ietf.org. A IN at zone org.
>>        [Got Referral to zone: ietf.org.]
>> Query: www.ietf.org. A IN at zone ietf.org.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.
>> Query: net. A IN at zone .
>>        [Got Referral to zone: net.]
>> Query: cloudflare.net. A IN at zone net.
>>        [Got Referral to zone: cloudflare.net.]
>> Query: cdn.cloudflare.net. A IN at zone cloudflare.net.
ERROR: NXDOMAIN: cdn.cloudflare.net. not found
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.

[Resolution stops here without yielding an answer]
```

NXDOMAIN answer at
empty non-terminal name

(Transcript from February 2015)

www.upenn.edu with minimization

```
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.

All of the following domains names are in the edgesuite.net. zone:

  edu-dscg.edgesuite.net.                    NXDOMAIN
  upenn.edu-dscg.edgesuite.net.              NXDOMAIN
  www.upenn.edu-dscg.edgesuite.net.          NOERROR with answer
```

The first two are both “empty non-terminals” (i.e. the domain name exists in the DNS tree, but has no data records associated with it, but has descendent names that do have data records).

The correct response for such names is NODATA, i.e. one with an empty answer section, response code of 0 (NOERROR), and the AA bit set.

Instead, response code 3 (NXDOMAIN) is returned.

NXDOMAIN is an authoritative indication that the queried name doesn’t exist in the DNS (and that nothing below it exists either).

www.upenn.edu with minimization

```
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.

   edu-dscg.edgesuite.net.              NXDOMAIN at ENT
   upenn.edu-dscg.edgesuite.net.        NXDOMAIN at ENT
   www.upenn.edu-dscg.edgesuite.net.    NOERROR with answer
```

Negative caching at intermediate qnames?

- For hosting providers that currently answer NXDOMAIN at empty non-terminals:
- Can we cause resolution failure of the target name by just querying the intermediate names?
  - By causing the resolvers to infer and cache the non-existence of everything beneath the empty non-terminals for a period of time?
  - (The negative cache TTL for edgesuite.net is 3 minutes).
- Answer: In most cases, no. Because current negative cache spec says resolvers should do **negative caching based on exact match of the qname**. Several popular resolvers I examined behave this way.
- But do all resolvers reliably follow this spec?

Negative caching of DNS responses

```
RFC 2308 (Negative Caching of DNS Queries (DNS NCACHE)

Section 5 says:

   A negative answer that resulted from a name error (NXDOMAIN) should
   be cached such that it can be retrieved and returned in response to
   another query for the same <QNAME, QCLASS> that resulted in the
   cached negative response.

   A negative answer that resulted from a no data error (NODATA) should
   be cached such that it can be retrieved and returned in response to
   another query for the same <QNAME, QTYPE, QCLASS> that resulted in
   the cached negative response.
```

But …

http://tools.ietf.org/html/draft-vixie-dnsext-resimprove-00
Vixie, Joffe, Neves, June 2010 (expired internet draft)

3. Stopping Downward Cache Search on NXDOMAIN

3.1. In virtually all existing resolvers, a cached NXDOMAIN is not considered "proof" that there can be no child domains underneath. This is due to an ambiguity in RFC 1034 that failed to distinguish empty nonterminal domain names from nonexistent names. For DNSSEC, the IETF had to distinguish this case, but the implication on non-DNSSEC resolvers wasn't fully realized.

3.2. When searching downward in its cache, an iterative caching DNS resolver should stop searching if it encounters a cached NXDOMAIN. The response to the triggering query should be NXDOMAIN.

3.3. When an iterative caching DNS resolver stores an NXDOMAIN in its cache, all names and RRsets at or below that node should be deleted since they will have become unreachable.

[…]

Larger scope of negative caching

- With minimization, resolution would necessarily stop when NXDOMAIN is encountered.
- This is beneficial because it allows negative caching to happen at higher layers of the DNS hierarchy, obviating the need to resolve many names under them.
- If “nxtld” doesn’t exist, then the resolver only needs to make a query for the first of blah1.nxtld., blah2.nxtld, blah3.nxtld, etc.

ENT NXDOMAIN workarounds?

- We expect Cloudflare, Akamai, etc. to address this problem before qname minimization becomes widespread.
- But, if this issue is widespread amongst many DNS providers, is it sensible to implement a workaround?
  - Ignore intermediate NXDOMAIN responses and continue pre-pending labels until the target qname is reconstructed?
- Probably not
  - Easy denial of service (DoS) attacks would be enabled.
  - Issue query for L1.L2.L3.L4….LN.<faultyzone>.
  - Cause resolvers to issue many unnecessary queries.

www.upenn.edu + NXDOMAIN workaround

```
$ resolve.py -mx www.upenn.edu. A
[…]
>> Query: www.upenn.edu. A IN at zone upenn.edu.
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.
>> Query: net. A IN at zone .
>>          [Got Referral to zone: net.]
>> Query: edgesuite.net. A IN at zone net.
>>          [Got Referral to zone: edgesuite.net.]
>> Query: edu-dscg.edgesuite.net. A IN at zone edgesuite.net.
ERROR: NXDOMAIN: edu-dscg.edgesuite.net. not found (ignoring)
>> Query: upenn.edu-dscg.edgesuite.net. A IN at zone edgesuite.net.
ERROR: NXDOMAIN: upenn.edu-dscg.edgesuite.net. not found (ignoring)
>> Query: www.upenn.edu-dscg.edgesuite.net. A IN at zone edgesuite.net.
www.upenn.edu-dscg.edgesuite.net. 21600 IN CNAME a1165.dscg.akamai.net.
>> Query: akamai.net. A IN at zone net.
>>          [Got Referral to zone: akamai.net.]
>> Query: dscg.akamai.net. A IN at zone akamai.net.
>> Query: a1165.dscg.akamai.net. A IN at zone akamai.net.
>>          [Got Referral to zone: dscg.akamai.net.]
>> Query: a1165.dscg.akamai.net. A IN at zone dscg.akamai.net.
www.upenn.edu. 300 IN CNAME www.upenn.edu-dscg.edgesuite.net.
www.upenn.edu-dscg.edgesuite.net. 21600 IN CNAME a1165.dscg.akamai.net.
a1165.dscg.akamai.net. 20 IN A 23.62.7.50
a1165.dscg.akamai.net. 20 IN A 23.62.7.64
```

(Transcript from February 2015)

www.ietf.org + NXDOMAIN workaround

```
$ resolve.py -mx www.ietf.org A
>> Query: org. A IN at zone .
>>         [Got Referral to zone: org.]
>> Query: ietf.org. A IN at zone org.
>>         [Got Referral to zone: ietf.org.]
>> Query: www.ietf.org. A IN at zone ietf.org.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.
>> Query: net. A IN at zone .
>>         [Got Referral to zone: net.]
>> Query: cloudflare.net. A IN at zone net.
>>         [Got Referral to zone: cloudflare.net.]
>> Query: cdn.cloudflare.net. A IN at zone cloudflare.net.
ERROR: NXDOMAIN: cdn.cloudflare.net. not found (ignoring)
>> Query: org.cdn.cloudflare.net. A IN at zone cloudflare.net.
WARNING: response REFUSED from 173.245.59.31
WARNING: response REFUSED from 2400:cb00:2049:1::adf5:3b1f
[.. Rest of REFUSED responses omitted for brevity ..]
ERROR: Queries to all servers for zone cloudflare.net. failed.
>> Query: ietf.org.cdn.cloudflare.net. A IN at zone cloudflare.net.
>> Query: www.ietf.org.cdn.cloudflare.net. A IN at zone cloudflare.net.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare.net.
www.ietf.org.cdn.cloudflare.net. 300 IN A 104.20.0.85
www.ietf.org.cdn.cloudflare.net. 300 IN A 104.20.1.85
```

(Transcript from February 2015)

www.ietf.org + NXDOMAIN workaround

The IETF/cloudflare canonical name is:

www.ietf.org.cdn.cloudflare.net.

Interestingly, the various empty non-terminals in this name exhibit a range of response behaviors:

```
cdn.cloudflare.net.                NXDOMAIN
org.cdn.cloudflare.net.            REFUSED
ietf.org.cdn.cloudflare.net.       OK (Empty, NOERROR)
```

Comments from Akamai, February 2015

Akamai has known about this issue for a while. Since this does not cause a problem in the real world today, it has not been fixed. But we are bumping the priority and have started internal discussions on resolving this. So we plan on addressing this but we do not yet have a firm schedule.

Comments from Cloudflare, February 2015

```
Cloudflare is aware of this defect and plans to fix it in
the near future.


Update: April 2015 from Cloudflare. The problem has been
fixed.
```

www.ietf.org after cloudflare DNSSEC (works!)

```
$ resolve.py -m www.ietf.org
>> Query: org. A IN at zone .
>>        [Got Referral to zone: org.]
>> Query: ietf.org. A IN at zone org.
>>        [Got Referral to zone: ietf.org.]
>> Query: www.ietf.org. A IN at zone ietf.org.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare-dnssec.net.
>> Query: net. A IN at zone .
>>        [Got Referral to zone: net.]
>> Query: cloudflare-dnssec.net. A IN at zone net.
>>        [Got Referral to zone: cloudflare-dnssec.net.]
>> Query: cdn.cloudflare-dnssec.net. A IN at zone cloudflare-dnssec.net.
>> Query: org.cdn.cloudflare-dnssec.net. A IN at zone cloudflare-
dnssec.net.
>> Query: ietf.org.cdn.cloudflare-dnssec.net. A IN at zone cloudflare-
dnssec.net.
>> Query: www.ietf.org.cdn.cloudflare-dnssec.net. A IN at zone cloudflare-
dnssec.net.
www.ietf.org. 1800 IN CNAME www.ietf.org.cdn.cloudflare-dnssec.net.
www.ietf.org.cdn.cloudflare-dnssec.net. 300 IN A 104.20.1.85
www.ietf.org.cdn.cloudflare-dnssec.net. 300 IN A 104.20.0.85
```

(Transcript from April 2015)

Brief survey of Alexa top 1000 domains

- Test performed on 2015-01-12 with the Alexa list of that date.
- Looked at both the bare domain and the bare domain with “www” prepended to it.
- Only queries for A records were performed (will try another pass with AAAA later).
- 1,986 such names exist.
- Attempted to lookup each of these names using the qname minimization algorithm.

Alexa 1000 survey results

```
#Domain names: 1986
#success:      1732   (87.2%)
#failure:       254   (12.8%)
```

Most failures are due to a small set of CDNs and DNS hosting providers.
Akamai is a dominant presence (identified by 3 of their known CDN domains: edgesuite.net, akadns.net, edgekey.net)

Summary of causes of unresolvability:

```
    184 akamai                           2 lxsvc.cn
     14 cdn20.com                        2 footprint.net
      8 incapdns.net                     2 <bad-referral>
      7 lxdns.com                        1 yhcdn.com
      7 chinacache.net                   1 nsatc.net
      6 ccgslb.com.cn                    1 netflix-cdn
      4 cloudflare                       1 deezer.com
      3 wscdns.com                       1 <circular-referral>
      3 <timeout>                        1 ccgslb.net
      3 <nodata>                         1 bitgravity.net
      2 tbcache.com
```

(Transcript from February 2015)

Partial list of failures (from top)

www.taobao.com, A, IN	tbcache.com
www.qq.com, A, IN	akamai
www.tmall.com, A, IN	tbcache.com
www.ask.com, A, IN	akamai
www.paypal.com, A, IN	akamai
www.apple.com, A, IN	akamai
www.microsoft.com, A, IN	akamai
www.gmw.cn, A, IN	ccgslb.com.cn
www.xinhuanet.com, A, IN	lxsvc.cn
www.163.com, A, IN	lxdns.com
www.netflix.com, A, IN	netflix-CDN
people.com.cn, A, IN	chinacache.net
www.people.com.cn, A, IN	chinacache.net
www.naver.com, A, IN	akamai
www.ebay.de, A, IN	akamai
www.cntv.cn, A, IN	lxdns.com
www.youku.com, A, IN	akamai
www.huffingtonpost.com, A, IN	akamai
www.chinadaily.com.cn, A, IN	chinacache.net
www.indiatimes.com, A, IN	akamai
www.dailymail.co.uk, A, IN	akamai
www.buzzfeed.com, A, IN	akamai
www.walmart.com, A, IN	akamai
pconline.com.cn, A, IN	<bad-referral>
www.pconline.com.cn, A, IN	cdn20.com
www.etsy.com, A, IN	akamai
www.yelp.com, A, IN	cloudflare

(Transcript from February 2015)

Qname minimization vs. query minimization

- Note that qname minimization does not mean minimization of the *number of queries.*
- With larger opportunities for negative caching, many queries for non-existent names will be eliminated.
- But, in many cases, minimization will cause additional queries to be issued by the resolver, e.g. in zones which contain names that are multiple labels deep. This is commonly the case in many leaf zones
- Caching delegation points and branch traversal depth will partially reduce the need for excess queries.
- An interesting study would be to try to measure and characterize the impact of these additional queries.

Additional queries at leaf zones

2 additional queries
at target zone

```
$ resolve.py -m www.net.isc.upenn.edu. A

>> Query: edu. A IN at zone .
>>        [Got Referral to zone: edu.]

>> Query: upenn.edu. A IN at zone edu.
>>        [Got Referral to zone: upenn.edu.]

>> Query: isc.upenn.edu. A IN at zone upenn.edu.
>> Query: net.isc.upenn.edu. A IN at zone upenn.edu.
>> Query: www.net.isc.upenn.edu. A IN at zone upenn.edu.
www.net.ISC.upenn.edu. 300 IN A 128.91.3.181
```

IPv6 reverse DNS example

```
Lookup reverse DNS entry for:
        www.seas.upenn.edu.    120  IN   AAAA 2607:f470:8:64:5ea5::9

$ resolve.py -m 9.0.0.0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.
7.0.6.2.ip6.arpa. PTR

>> Query: arpa. PTR IN at zone .
>> Query: ip6.arpa. PTR IN at zone .
>>          [Got Referral to zone: ip6.arpa.]
>> Query: 2.ip6.arpa. PTR IN at zone ip6.arpa.
>> Query: 6.2.ip6.arpa. PTR IN at zone ip6.arpa.
>> Query: 0.6.2.ip6.arpa. PTR IN at zone ip6.arpa.
>>          [Got Referral to zone: 0.6.2.ip6.arpa.]
>> Query: 7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>>          [Got Referral to zone: 0.7.4.f.7.0.6.2.ip6.arpa.]
>> Query: 0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.

[ … stream of queries continues … ]
```

3 queries

5 queries

The leaf zone (2607:f470::/32)
24 queries (i.e. 23 extra!!)

IPv6 reverse DNS example

```
$ resolve.py -m 9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR
>> Query: arpa. PTR IN at zone .
>> Query: ip6.arpa. PTR IN at zone .
>>        [Got Referral to zone: ip6.arpa.]
>> Query: 2.ip6.arpa. PTR IN at zone ip6.arpa.
>> Query: 6.2.ip6.arpa. PTR IN at zone ip6.arpa.
>> Query: 0.6.2.ip6.arpa. PTR IN at zone ip6.arpa.
>>        [Got Referral to zone: 0.6.2.ip6.arpa.]
>> Query: 7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>> Query: 0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.6.2.ip6.arpa.
>>        [Got Referral to zone: 0.7.4.f.7.0.6.2.ip6.arpa.]
>> Query: 0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 0.0.0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
>> Query: 9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.a.e.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. PTR IN at zone 0.7.4.f.7.0.6.2.ip6.arpa.
9.0.0.0.0.0.0.0.0.0.0.0.0.0.0.5.A.E.5.4.6.0.0.8.0.0.0.0.7.4.f.7.0.6.2.ip6.arpa. 120 IN PTR www.seas.upenn.edu.
```

What next?

- Measurements across larger set of domains.
- Implement query-type-hiding version of algorithm.
- Replay query trace from a large resolver.
- Analysis of number of queries issued compared to the normal iterative resolution algorithm.
  - At a defined set of target domains
  - Using query traces at large resolvers
  - Using query traces at root/com/net authoritative servers
- Contact other CDNs and DNS hosting providers exhibiting problems.

Parting thoughts

- DNS privacy is important.
- Qname minimization is one dimension of that privacy.
- Some nameserver implementations will need to be fixed.
- Query volume impacts should be studied.
- We should think about loss of information at higher layers of the DNS that has traditionally been used for a variety analytical and security functions.
    - "What Information does a Name Server need to do its job?" – Burt Kaliski
        - http://blogs.verisigninc.com/blog/entry/minimum_disclosure_what_information_does

Questions or comments?

Shumon Huque <shuque @ verisign.com>

powered by

VERISIGN™