AdaDoQ: Adaptive DNSSEC

Yehuda Afek
Tel-Aviv University

Anat Bremler-Barr
Reichman University

Daniel Dubnikov
Tel-Aviv University

Supported By:
Motivation

- DNSSEC is important
- DNS with DNSSEC does not scale, specifically, ➔ Vulnerable to NXDomain flood attacks

Goal

1. To measure DNSSEC scalability relative to Plain DNS
2. Develop a method for <resolver ↔ authoritative> collaboration that is
   (a) Scalable, (b) as secure as DNSSEC, and (c) introduces no new vulnerabilities.

   a. Provides the same security level as DNSSEC, and
   b. Provides performances close to that of Plain-DNS, and
   c. Does not enable new vulnerabilities.
Outline

• Motivation: DNSSEC under NXDomain flood attack
  • Either slow or Zone walking vulnerability

• AdaDoQ: solution:
  • TCP/TLS
  • DNSSEC PKI hierarchy of trust
  • QUIC

• AdaDoQ performances

• Conclusions
NXDomain Request

Client

Recursive Resolver

Empty cache

fake.example.com ??

nx-domain fake.example.com

Empty cache

1. fake.example.com ??
2. Ref NS for .com
3. fake.example.com ??
4. Ref NS for example.com
5. fake.example.com ??
6. nx-domain fake.example.com

root

a.root-servers.net

TLDs

com .edu .us .net

SLDs

ns.example.com
With DNSSEC

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
</tr>
</tbody>
</table>

Client
fake.example.com ??
Empty cache

Recursive Resolver

nx.example.com
Empty cache

Root
a.root-servers.net

TLDs
.com .edu .us .net

SLDs

Under NXDomain attack
NXDomain Attack

RANDOM DNS Request Flood

Resolvers

- Rxy1xhggsgVCER.sony.com
- XVBY$&HGDRxy2.sony.com
- FJH*^DHGAKRxy3.sony.com
- RxUYQVMNLKAy4.sony.com
- RzHW$RE43CBJs$7.sony.com

ns.sony.com

.com
Motivation (1) NSEC  aggressive caching

• Non Existent query in DNSSEC?
Motivation (2)

• Non Existent query in DNSSEC?
• Query: ddd.name.com?
Motivation (2) - NSEC record

- Non Existent in DNSSEC?
- Query: ddd.name.com?
- **NSEC** Record:

Query: Junk.NAME.COM
Nothing exists in-between
Motivation (2) NSEC aggressive caching

- Aggressive Caching (RFC 8198) – stops NX Attack

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive Caching</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>96,226</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>93,756</td>
</tr>
</tbody>
</table>

With Knot
Motivation (2) NSEC aggressive caching

- Aggressive Caching (RFC 8198) – stops NX Attack

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
</tr>
<tr>
<td>Aggressive Caching</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>96,226</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>93,756</td>
</tr>
</tbody>
</table>
Motivation (3) Zone walking

- Aggressive Caching (RFC 8198) – stops NX Attack
- **BUT:** Enables Zone Walking

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive Caching</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>96,226</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>93,756</td>
</tr>
</tbody>
</table>

With Knot
Motivation (3) Zone walking

- Aggressive Caching (RFC 8198) – stops NX Attack
- **BUT**: Enables Zone Walking
  
  Scalability Issues: Need to quickly find NSEC record

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive Caching</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>96,226</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>93,756</td>
</tr>
</tbody>
</table>
Motivation (3) Zone walking

• How to stop Zone Walking?
Motivation (3) NSEC3 and Zone walking

• How to stop Zone Walking?

• NSEC3 hash of the interval (the domain names)
  • Keep the hash function on both sides.
  • Aggressive caching still works. Zone walking is harder, But
  • Still can do Zone Walking with dictionary attack:
    An attacker collects all the NSEC3 records, and uses dictionary attack to reveal
    the true domain-names
Motivation (3) stop Zone walking by Black/White lies

- You can’t (without online signing*, Goldberg et al.)
  Nsec5: Provably preventing dnssec zone enumeration

- **Black Lies, White Lies, NSEC5**

_query: Junk.NAME.COM_

aaa.Name → Index.NAME.COM → Mail.NAME.COM → Junk.NAME.COM - Junk.NAME.COM

- Must sign the record on the fly/online
Motivation (4)

- Online Signing Algorithms – NX Attacks Amplified

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
<th>% of Plain DNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
<td>100%</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
<td>40%</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
<td>38%</td>
</tr>
<tr>
<td>DNSSEC: White Lies</td>
<td>5,863</td>
<td>25%</td>
</tr>
<tr>
<td>DNSSEC: Black Lies</td>
<td>7,206</td>
<td>30%</td>
</tr>
<tr>
<td>DNSSEC: NSEC5</td>
<td>6,324</td>
<td>27%</td>
</tr>
</tbody>
</table>
Motivation (4)

• Online Signing Algorithms – NX Attacks Amplified

• For security and scalability reasons online signing might be the only option

<table>
<thead>
<tr>
<th></th>
<th>Max Queries Per Second</th>
<th>% of Plain DNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
<td>100%</td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
<td>40%</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
<td>38%</td>
</tr>
<tr>
<td>DNSSEC: White Lies</td>
<td>5,863</td>
<td>25%</td>
</tr>
<tr>
<td>DNSSEC: Black Lies</td>
<td>7,206</td>
<td>30%</td>
</tr>
<tr>
<td>DNSSEC: NSEC5</td>
<td>6,324</td>
<td>27%</td>
</tr>
</tbody>
</table>
DNSSEC

- Increase packet size
- Increase response count
- Add CPU load
  -> DoS Amplification

W/B Lies + NSEC5

NSEC/3 Aggressive Caching

- Zone Walking
- Scalability (for CDNs)
- Disposable Domains
Proposed Solution - Hybrid

- **Hybrid Resolver**
  - **Authoritative Nameserver**
  - **DNS over Secure TLS**
  - **DNSSEC over UDP**

- **Heavy load**
  - **DNS over TLS**
  - **DNSSEC over UDP**

- **Light load**
  - **DNS over Secure TLS**
  - **DNSSEC over UDP**

- **High Traffic** -> **DNS over Secure TLS**
- **Low Traffic** -> **DNSSEC over UDP**

- **Authoritative**
  - Authenticates the resolver once
  - Following traffic is sent without DNSSEC signatures and is considered validated
Proposed Solution (1)

• Remove all DNSSEC overheads
  • Packet Size + extra packet (NSEC record)
  • CPU Load
Proposed Solution (1) TLS

• Remove all DNSSEC overheads
  • Packet Size + Count
  • CPU Load

• Use TLS
  • Resolver initiates
  • Authoritative identifies itself once,
  • Traffic sent with Plain DNS over TLS.
  • **Using **DNSSEC hierarchical chain of trust ** instead of TLS certificates.**
  • **TLS == DNSSEC**: Only owner of DNSSEC key can authenticate information
Proposed Solution – PoC (1) Proxy

• Problem1: Can’t easily integrate with known resolver/auth implementations (Bind, Unbound, Knot, etc.)

• Solution: proxy interface resolver - authoritative servers
Proposed Solution – PoC (2) QUIC

• **Problem2:**
  • TLS overhead is high
  • TLS suffers from Head-of-Line blocking
Proposed Solution – PoC (2) QUIC

- **Problem 2:**
  - TLS overhead is high
  - TLS suffers from Head-of-Line blocking

- **Solution:** Use **QUIC** (similar to HTTP3)
  - Over UDP
  - UDP Multiplexing (virtual connections) ➔ No head of the Line Blocking
  - Connection kept open: resume connection with 0 round trip time
  - No need for TCP integration (firewalls/IPS)
  - At most one QUIC connection for each Resolver-Authoritative pair.
    QUIC anti-spoofing protection
Proposed Solution – PoC (3)

- **Problem3**: Teardown and restart QUIC connections
Proposed Solution – PoC (3) Long lived

• **Problem3**: Teardown and restart QUIC connections

• **Solution**: Keep connections alive
  - QUIC has low overhead – long lived connections
  - QUIC can resume quickly
Proposed Solution – PoC (4)

• Problem4: Resource limit
Proposed Solution – PoC (4) Limit # QUIC Connections

• **Problem**: Resource limit

• **Solution**: Score connection throughput with

\[
F_i = \alpha \cdot F_{i-1} + (1 - \alpha) \cdot f_i
\]

- Resolver terminates lowest scored connection (LRU)
- Close connection when \( F_i \) is below a lower threshold
Proposed Solution – PoC (5) Attack Tolerance

• **Problem5:** Estimate impact of resource exhaustive attacks

• Still need to perform measurement. However:
  
  ➔ Over UDP instead of TCP

  ➔ Resolver terminates lowest scored connection (LRU).

  ➔ Threshold for closing connection lower than opening threshold
Proposed Solution – PoC (6) No DNSSEC validation

• **Problem6**: Clients cannot validate DNSSEC signatures.

→ Trust the resolver.
<table>
<thead>
<tr>
<th>Knot</th>
<th>Max Queries Per Second</th>
<th>% of Plain DNS</th>
<th>ECDSA P-256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
<td>40%</td>
<td>4,213</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
<td>38%</td>
<td>4,015</td>
</tr>
<tr>
<td>DNSSEC: White Lies</td>
<td>5,863</td>
<td>25%</td>
<td>2,070</td>
</tr>
<tr>
<td>DNSSEC: Black Lies</td>
<td>7,206</td>
<td>30%</td>
<td>3,338</td>
</tr>
<tr>
<td>DNSSEC: NSEC5</td>
<td>6,324</td>
<td>27%</td>
<td>2,171</td>
</tr>
</tbody>
</table>
Measurements

With QUIC, using the same experiment (NX flood), throughput is **87%** of the plain DNS.

<table>
<thead>
<tr>
<th>Knot</th>
<th>Max Queries Per Second</th>
<th>% of Plain DNS</th>
<th>ECDSA P-256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plain DNS</td>
<td>23,524</td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td>DNSSEC: NSEC</td>
<td>9,510</td>
<td>40%</td>
<td>4,213</td>
</tr>
<tr>
<td>DNSSEC: NSEC3</td>
<td>8,989</td>
<td>38%</td>
<td>4,015</td>
</tr>
<tr>
<td>DNSSEC: White Lies</td>
<td>5,863</td>
<td>25%</td>
<td>2,070</td>
</tr>
<tr>
<td>DNSSEC: Black Lies</td>
<td>7,206</td>
<td>30%</td>
<td>3,338</td>
</tr>
<tr>
<td>DNSSEC: NSEC5</td>
<td>6,324</td>
<td>27%</td>
<td>2,171</td>
</tr>
<tr>
<td>AdaDoQ (Our Solution)</td>
<td>20,558</td>
<td><strong>87%</strong></td>
<td></td>
</tr>
</tbody>
</table>
\[ \begin{align*}
\text{Less secure} & : \text{NSEC3, NSEC} \\
\text{More secure} & : \text{NSEC5} \\
\text{Drawbacks} & : - \text{Slow Update} \\
& \quad - \text{Hot ZSK} \\
& \quad - \text{Zone Enumeration} \\
& \quad - \text{Poisoning} \\
& \quad - \text{Hijacking} \\
\text{Aggressive Caching} & : \text{AdaDoQ & BL} \\
\text{NX QPS} & : 5.8k, 6.3k, 7.2K, 9K, 9.5K, 13.9K, 14.7K, 20.5K, 23.5K, 96K \\
\end{align*} \]
Online Signing

NSEC5

White Lies
Black Lies

Offline Signing

NSEC3
NSEC

Drawbacks

- Slow Update
- Hot ZSK
- Zone Enumeration

- Poisoning Hijacking

Less secure

More secure

Aggressive Caching

Plain DNS

NSEC3
NSEC

Better

9K 9.5K 13.9K 14.7K 20.5K 23.5K 96K

Poisoning Hijacking
Zone Enumeration
Hot ZSK

9K

13.9K

6.3K

-2K

2K

14.7K

9.5K

5.8K

298

.5K

14.7K

13.9K

6.3K

-2K

2K
Conclusions

• DNSSEC degrades DNS performance
  • Make NXDOMAIN attacks worse (DDoS amplification)

• AdaDoQ – Hybrid Solution
  • Light and fast connections
  • One time encryption overheads
  • No Security Compromises
  • No Zone Walking
  • Close to Plain DNS throughput
  • No Scalability Issues
Questions?