DNS Security

Wes Hardaker  <hardaker@isi.edu>
Wildcards -- ERRATA (RFC4592)

- Generating responses for missing data
  - Left most label must be a "*" (and only a "*")
  - Matches any label that doesn’t already exist
    - Including sub-labels under it
  - Causes a nameserver to *synthesize and answer*
  - Please read RFC4592! Good examples therein.

- Example records:

  
  * example.com. 3600 IN MX 10 mail.example.com
  host1.example.com. 3600 IN A 192.0.2.1

- Responses:

  host1.example.com/MX  
  host2.example.com/MX  
  host1.example.com/A  
  host2.example.com/A

  
  *host1.example.com/MX  
  host2.example.com/MX  
  host1.example.com/A  
  host2.example.com/A

  DOESN’T MATCH (returns NOERROR)
  MATCHES
  DOESN’T MATCH (returns 192.0.2.1)
  DOESN’T MATCH (returns NOERROR)
DNS Security
DNS Attack Surfaces

Like every other early protocol, DNS started out insecure.
DNS Publication Architecture

DNS Zone Content
Starts Here

DNS Users
and applications

Queried by Recursive Resolvers
(ISPs, Quads, etc)

Data Served By
Authoritative DNS Servers
(ns0.amsl.com)

Registries
(PIR: .org)

Registrars
(Network Solutions)

Zone Owner
Registrant
(ietf.org)

Original diagram from: Russ Mundy
Sources of DNS Insecurity: Registration

**Question:** How do you get bad data into the DNS itself?

**Answer:** Insert it during publication side

Example methods of attack:

1. Social engineering at the publishing side
   - a. Registry attacks
   - b. Registrar attacks

2. Attack the zone source itself
   - a. Security protocols won’t help you if your source data isn’t stored securely!
Sources of DNS Insecurity: Publication / Consumption

1. Protocol attacks
   a. Cache Poisoning

2. DDoS attacks: target and participant
   a. Resource consumption
   b. Reflection
   c. Loops

3. Privacy
   a. Queries are seen by both resolvers and authoritative servers

4. DNS abuse and deception
   a. One-off names/typos/bits and (ietf -> ieetf, itf, ietff, ietforg)
   b. Internationalization and other font issues (IETF -> 1ETF)
DNS Architecture Refresher

Resolvers query the tree to find the answer you need

www.example.com?
Cache Poisoning

Without security, First answer wins

www.example.com?

www.example.com is at 6.6.6.6!

Resolver
Early and Easiest Cache Poisoning Attacks

- Extra data in the additional section was considered authentic
  - I know you asked for www.example.com,
  - but did you know www.ietf.org is at 6.6.6.6?
    ■ It is! And you should trust me!

- Protocol attacks
  - Just flood the DNS resolver with response packets and hope they accept them
  - Possibly guessing IP/protocol values (name, source address, DNS ID field)
  - Answer is cached for TTL chosen by the attacker
Early and easiest methods to combat cache poisoning

- Ignore non-authoritative answers
  - Only accept answers to questions you asked
  - Exception: parents can supply “glue” address records for in-zone child NS servers

- Resolvers must check that:
  - The IP source and UDP port is correct (handled by the UDP stack)
  - The DNS ID field is correct

- Senders must make it harder for attackers to guess these
  - Randomize the source port number
  - Randomize the ID field

- Note that this
  - Isn’t cryptographically strong
  - Doesn’t work at all for on-path attackers that can see and copy the requests

32 bits of randomness
But can we fix it?

Yes we can!
Data protection mechanisms

Object security

Path security
Object Security -- DNSSEC

- Adds cryptographic signatures on records
- Signed at or near the data’s origin
- Verifiable in the middle
- Verifiable at the end
- Only provides integrity protection -- no privacy protection
Path Security -- DOT, DOH, DOQ, ...

- Tunnel’s answers securely between two points
- Provides integrity protection and Encryption
  - But, offers point-to-point protection only
  - Verifies **who you got it from**, but **not what it is**
Object vs Path Security Compatibility

They have very different complementary properties
DNS Security Technologies

Object security (integrity, but no encryption)

- DNSSEC: DNS records signed by their authoritative source. RFC4033+

Path security (point-to-point integrity, and provides encryption)

- DoT: DNS over TLS. RFC7858
- DoDT: DNS over DTLS. RFC8094
- DoH: DNS over HTTPS. RFC8484
- ODoH: Oblivious DoH. DNSOP
- TSIG: DNS over DNS with shared keys. RFC8945
- DNS Curve: point-to-point encryption and authentication with elliptic curve
Security Protection Areas

- **www.example.com?**
  - Resolver
  - DoT / DoDT / DoH
  - Authenticates and Encrypts the user to the resolver's protocol traffic

- **DNSSEC**
  - Authenticates the resource records inside the protocol traffic

- **root**
  - com
  - org
  - example
  - ietf
  - icann
  - www
  - ns

Can be used in both places, but typically isn’t
DNSSEC secures records no matter where they are
DNSSEC - hierarchical object signing security

www.example.com?

Resolver

AD=1
All good!

Problem: this message is typically sent insecurely

Signs root
Signs com’s DS

Signs com
Signs example.com’s DS

Signs example.com
Signs www.example.com

root

com

org

ietf

icann

www

ns

public/private
Key pairs

Signs root
Signs com’s DS

Signs com
Signs example.com’s DS

Signs example.com
Signs www.example.com

DNSKEY

DS

RRSIG

DS

DNSKEY

RRSIG

DS

DNSKEY

RRSIG

A
Path vs Object Differences

Object security  
(DNSSEC)

- **Pro:** Ensures end-to-end integrity is available everywhere
- **Con:** Doesn’t provide privacy protection
- **Pro:** Distributed trust model with minimal configuration (typically 1 anchor)
- **Con:** Typically not deployed all the way to the user (“the last mile”)

Path security  
(DoT, DOH, …)

- **Pro:** Provides privacy protection and point-to-point integrity protection
- **Con:** Doesn’t verify data actually came from the origin (trust everyone?)
- **Con:** For true security, requires that every link be protected & trusted
- **Pro:** Solves the last mile problem
Path and Object security

Stronger together:

Object security protects the data
Object security ties the date to its source
Path security fixes the last mile problem
Path security provides encryption

Protects clients from on-path eavesdropping
DNS as a source of DDoS Attacks

Because:

- DNS primarily uses the spoofable UDP protocol, and
- Most DNS responses are larger than the requests

Leads to:

- DNS is used in reflection attacks
DNS as a DDoS Target

Because:

- Nearly all Internet communication starts with DNS lookups

Leads to:

- DNS itself is often a DDoS target
To battle aggressive clients: deploy RRL

Dear client: Why do you ask so many questions?

Solution: Deploy Response Rate Limiting (RRL)

- I’ve answered you N times already in the last M minutes
- As punishment:
  - I’m setting the Truncated Bit
  - Please come back using TCP please (RFC7766)
  - … or implement DNS cookies (RFC7873)

Helps prevent overwhelming spoofed UDP requests

Note: Not-standardized, but widely implemented and deployed
To battle DNS outages: Optimize your caches

These all provide some resilience against network issues and DDoS attacks:

Serve Stale

TL;DR: If you can’t get a new answer, it’s ok to use an old one

Running a copy of the root on your local resolver

TL;DR: Don’t ask questions you know there isn’t an answer for

NSEC Aggressive Caching

Reduces traffic loads to authoritative servers
Additional Security Topics Not Covered

- It’s possible to have multiple trust anchors
- NSEC3
- Bootstrapping security using DNSSEC signed public keys
  - DANE, PGP, SSHFP, ...
- Infrastructure interdependency concerns
- Deployment levels
- Algorithm strengths and rollovers
- Replay attacks and solutions
- QName Minimization

- Query loops
- Unauthorized authoritative servers
- DNS intercepting middle-boxes
- White-lies & black-lies
- Minimal signing
- Many new things in WG states