Introduction to DNSSEC & DANE

Josh Kuo

josh@deepdivenetworking.com
DeepDive Networking possesses and delivers an incredible depth of knowledge related to DNS, DHCP, and core networking technologies. Our core specialties include, architecture, design, implementation services, training delivery, and training development for firms worldwide. DeepDive strives to exceed all expectations, and delivers master-level results to ensure ultimate, repeatable success.

http://www.deepdivenetworking.com
What Are We Talking About?

We are talking about 3 things basically:

1. What is DANE?
2. Why is DNSSEC necessary for us to use cool things like DANE?
3. How does DNSSEC work?

Everything I am about to talk about here is open standards, nothing proprietary, share it!
What is DANE?

**DNS-based Authentication of Named Entities**

RFC 6698 (August 2012)
RFC 7218 (April 2014)

Basically, DANE allows us to store information about generic crypto objects such as a X.509 certificate (commonly known as SSL/TLS certs) in DNS as a TLSA record, it looks like this:

```
_443._tcp.www.mydnsseccgood.org. 3600 IN TLSA 3 0 1
85E4C96EA373020E6B558F657F61DD275E5FBD649280A3A7A0A848D4 ED8457C9
```
Uses of DANE

1. Use DANE as a verification mechanism to verify SSL/TLS certificates received over HTTPS for added security

2. Store self-signed X.509 certificates, bypass having to pay a third party*

3. Integrate with Mail Transfer Agents (MTA) to provide seamless, end-to-end email encryption

* Requires smarter applications
# Why Verify Certs?

Don’t we trust Certificate Authorities (CA)?

### Certificate Authorities:

<table>
<thead>
<tr>
<th>Certificate Name</th>
<th>Security Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>(c) 2005 TÜRKTRUST Bilgi İletişim ve Bilgi Güvenliği Hizmetleri A.Ş. TÜRKTRUST Elektronik Sarti'sı Hizmet Sağlayıcı</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>A-Trust Ges. v. F. Sicherheitsysteme im elektr. Datenverkehr GmbH</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AC Camerimma S.A.</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Chambers of Commerce Root - 2006</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Global Chambersign Root - 2008</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AC Camerimma SA CIF A82743287</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Chambers of Commerce Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Global Chambersign Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>ACCV</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>ACCVFAZ1</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Actalis S.p.A./03358520967</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Actalis Authentication Root CA</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AdTrast AB</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AdTrast External CA Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AdTrast Class 1 CA Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AdTrast Public CA Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>AdTrast Qualified CA Root</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>COMODO High-Assurance Secure Server CA</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>PositiveSSL CA 2</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>COMODO SSL CA</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>COMODO RSA Certification Authority</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>COMODO SSL CA 2</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>USERTrust Legacy Secure Server CA</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>UTN - DATACorp SGC</td>
<td>Software Security Device</td>
</tr>
</tbody>
</table>
Why Verify Certs?

But if a certificate is “known bad”, we can revoke it, right? Surely our browsers will check that for us, right? Right?
DANE Verification Overview

HTTPS request for www.example.com

Here's my certificate, trust me!

DNS Server, what is the TLSA record for www.example.com?

Mismatch! You are an impostor!!

Answer: 3 0 1 85E4C96EA373020E6B558F657F61DD275E5FBD649280A3A7A0A848D4 ED8457C9
Self-Signing Certificate with DANE

Limited support today:

- Firefox with a plugin
- Bloodhound Browser (Mozilla)

Resources:

http://users.isc.org/~jreed/dnssec-guide/dnssec-guide.html#recipes-tlsa
http://dane.verisignlabs.com
https://www.dnssec-validator.cz/
http://www.ietf.org/mail-archive/web/dane/current/pdfk2DbQF0Oxs.pdf
Automatic Email Encryption with DANE

- Leveraging DANE, MTA (email server) can encrypt an email before it is sent on the wire.
- Postfix 2.11.1 supports opportunistic encryption using OpenPGP keys published in DNS as TLSA records.
- Still in draft status.

Other Similar Record Types

Other DNS Resource Records that work similarly to DANE (TLSA):

1. SSHFP (RFC 4255)
2. IPSECKEY (RFC 4025)
3. TXT Record (Spam Detection):
   1. SPF (http://www.openspf.org/)
   2. DKIM (http://www.opendkim.org/)
   3. DMARC (http://dmarc.org)
example.com. 3600 IN TXT "v=spf1 mx ip4:45.0.0.0/15 -all"

`spf1` = SPF version
`mx` = whatever I have listed in my MX records
`ip4:45.0.0.0/15` = email from this network is ok
`-all` = fail everyone else
Here’s email from example.com, my IP is 6.6.6.6

DNS Server, what is the TXT record for example.com?

Answer: "v=spf1 mx ip4:45.0.0.0/15 -all"

You don’t check out, SPAM!!!
Hey, I can store that in...

F**k it, it’s in DNS now.
But unlike CAs, we can totally trust DNS, right?

Question: What is the IPv6 address of www.example.com?

Answer: 2607:dead:beef:cafe::68
## Trusting DNS

### DNS Header

<table>
<thead>
<tr>
<th>Field</th>
<th>Structure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>1 byte</td>
<td>Identifies the DNS protocol version</td>
</tr>
<tr>
<td>Header Length</td>
<td>2 bytes</td>
<td>Length of the DNS header</td>
</tr>
<tr>
<td>Identifier</td>
<td>2 bytes</td>
<td>Unique identifier for each query</td>
</tr>
<tr>
<td>Type</td>
<td>1 byte</td>
<td>Type of query (e.g., A, AAAA)</td>
</tr>
<tr>
<td>Class</td>
<td>1 byte</td>
<td>Class of resource (e.g., IN, CH)</td>
</tr>
<tr>
<td>TTL</td>
<td>4 bytes</td>
<td>Time to live (number of seconds)</td>
</tr>
<tr>
<td>Protocol</td>
<td>2 bytes</td>
<td>Protocol number (e.g., DNS is 206)</td>
</tr>
<tr>
<td>Source IP</td>
<td>4 bytes</td>
<td>Source IP address of the query host</td>
</tr>
<tr>
<td>Destination IP</td>
<td>4 bytes</td>
<td>Destination IP address of the response</td>
</tr>
<tr>
<td>Source Port</td>
<td>2 bytes</td>
<td>Source port number</td>
</tr>
<tr>
<td>Destination Port</td>
<td>2 bytes</td>
<td>Destination port number</td>
</tr>
<tr>
<td>UDP Length</td>
<td>2 bytes</td>
<td>Length of the UDP payload</td>
</tr>
<tr>
<td>UDP Checksum</td>
<td>2 bytes</td>
<td>Checksum of the UDP payload</td>
</tr>
<tr>
<td>Query ID</td>
<td>2 bytes</td>
<td>Unique identifier for each query</td>
</tr>
<tr>
<td>Question Count</td>
<td>2 bytes</td>
<td>Number of questions in the query</td>
</tr>
<tr>
<td>Answer Count</td>
<td>2 bytes</td>
<td>Number of answers in the response</td>
</tr>
<tr>
<td>Authority Count</td>
<td>2 bytes</td>
<td>Number of authorities in the response</td>
</tr>
<tr>
<td>Additional Count</td>
<td>2 bytes</td>
<td>Number of additional records in the response</td>
</tr>
</tbody>
</table>

### DNS Data

- **DNS question** or **answer data**

### Security Features

- **DNSSEC** ensures the authenticity of DNS records.
- **DANE** (Domain-based Authentication of Named Entities) provides a way to verify that a DNS response came from a trusted source.

---

**DeepDive Networking**

**DNSSEC & DANE**
Question: What is the IPv6 address of www.example.com?

Answer: NXDOMAIN

QID: 11116

Answer: 2607:dead:beef:cafe::77

QID: 11116
DNSSEC provides:
1. Authentication
2. Data Integrity
3. Proof of non-existence

RFC 4034, 4034, and 4035 outline the basics
Uses public key crypto and digital signatures

But not data privacy, no encryption!
1 Question: What is the IPv6 address of www.example.com?

3 Question: I need keys to read signatures on child example.com.

5 Question: Give me key info on child example.com.

7 Question: Your keys please, root zone.

9 Question: Give me key info on your child com.

11 Question: Your keys please, root zone.

2 Answer: Here is the IPv6 address and signature(s).

4 Answer: keys for you to read signatures from #2.

6 Answer: Key info that child uploaded to me, go verify #6, you can verify #4.

8 Answer: My keys, which you can use to verify #6.

10 Answer: Key info com uploaded to me, go verify #8.

12 Answer: Here are my keys.
Network Requirements for DNSSEC

- DNS server supports EDNS0 (large UDP packets)
- Network gear *not* drop large DNS packets (larger than 1500 bytes typically)
- Network is aware of DNS over TCP
How Do I Know I Have DNSSEC?

- Recursive servers, look for **ad** flag in returned header (ad = authenticated data)
  
  `dig @4.2.2.2 www.isc.org. A`
  `dig @8.8.8.8 www.isc.org. A`

- Authoritative servers, use `dig +dnssec`
  
  `dig enet.interop.net. SOA +dnssec`
  
That’s right, Google has been providing DNSSEC validation since 2013.
Challenges of DNSSEC

- Perception: it’s DNS with crypto, it’s hard!
- It will break lookups! (8.8.8.8)
- Does not solve last mile problem (yet)
- No incentives, maybe PCIDSS will fix that
- We need to reach critical mass like .gov
- DANE working group
  https://datatracker.ietf.org/wg/dane/
- DNS Private Exchange working group
  http://datatracker.ietf.org/wg/dprive/