Ch 6: DNSSEC and Beyond

Updated 11-22-16
DNSSEC
Objectives of DNSSEC

• Data origin authentication
  – Assurance that the requested data came from the genuine source

• Data integrity
  – Assurance that the data have not been altered
Development of DNSSEC

• Record signatures use public-key cryptography to verify authenticity of DNS records
• RFC 2065 (1997): SIG and KEY records
• RFC 2535 (1999): NXT record - denial of existence of a record
• 2003: DS (Delegation Signer) record
  – Allows secure delegation to child zones
  – SIG, KEY, NXT evolved to RRSIG, DNSKEY, NSEC
Development of DNSSEC

• RFC 3757 (2004)
  – Zone-Signing Key (ZSK)
  – Key-Signing-Key (KSK)
  – Secure Entry Point (SEP)
    • A flag used to identify a KSK

• 2005
  – NSEC replaced by NSEC3
Figure 53: The process of signing the resource records of a DNS zone. Using a public key cryptography algorithm, we first generate a key pair, consisting of a private and a public key. A set of records (RR set) are passed through a hash function (such as SHA1 or SHA-256). The private key is then used to sign the hash of the selected resource records using an algorithm such as RSA, DSA or Elliptic Curve. The resource records, along with their corresponding signature and public key are published in the zone file.
Man-in-the-Middle Attack

- Attacker generates two “false” key pairs
- Attacker intercepts the genuine keys and send false keys out
- Client trusts the Attacker's data

• Trusted third party prevents this attack
  – The root of DNS
Hierarchical Chain of Trust
The Delegation Signer (DS) Record

• Links in the chain of trust
• DS record contains a hash of a zone's public key
• To make performance better, the zone key may be separated into
  – Zone Signing Key (ZSK) and
  – Key Signing Key (KSK)
Figure 56: Delegation Signer (DS) records are used to establish a chain of trust in DNSSEC between a parent and a child.
Authenticated Denial of Existence
NSEC and NSEC3 Records

• Sort all domain names in zone in alphabetical order
  – a.example.com with A, AAAA, RRSIG
  – c.example.com

• NSEC Record
  – a.example.com TTL IN NSEC c.example.com A AAAA RRSIG
  – Nothing between a.example.com and c.example.com

• This record proves there is no b.example.com
Example

~ $dig a.b.c.d.us. +dnssec

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>us.</td>
<td>21599</td>
<td>IN</td>
<td>NSEC</td>
<td>0-.us.</td>
<td>NS</td>
<td>SOA</td>
<td>RRSIG</td>
</tr>
</tbody>
</table>

- Proves there is no *.us

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CZZH.us.</td>
<td>21599</td>
<td>IN</td>
<td>NSEC</td>
<td>D-.us.</td>
<td>NS</td>
<td>RRSIG</td>
<td>NSEC</td>
</tr>
</tbody>
</table>

- Proves there is no d.us
NSEC Information Disclosure

• Easy to find all domains in a zone
• Like a zone transfer
• `dig *.se +dnssec`
  – Reveals first valid hostname
• Dig for valid hostname* to find next one
• NSEC3 fixes this problem
NSEC3 Hostnames are Hashed and Salted

- Can be hashed many times
Algorithm and Salt in Record

The presentation format of the record also contains a hash algorithm identifier (e.g., SHA1 etc.), a flags field, the number of hashing iterations executed, and the salt used to seed the hash:

```
<Hashed_owner_name.zonename>  <TTL>  <CLASS>  NSEC3
<Algorithm ID>  <Flags>  <Iterations>  <Salt>
<Next_hashed_owner_name>  <List of Types at the original owner name>
```
dig 9.8.3-P1 a.b.cia.mil +dnssec

;; global options: +cmd
;; Got answer:
;; ->>>HEADER<<< opcode: QUERY, status: NXDOMAIN, id: 2162
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 1

;; OPT PSEUDOESECTION:
;; EDNS: version: 0, flags: do; udp: 512

;; QUESTION SECTION:
:a.b.cia.mil. IN A

;; AUTHORITY SECTION:

mil. 1000 IN SOA CON1.NIPR.mil. DISA\COLUMBUS\NS\MBX\HOSTMASTER-DOD-NIC.

MAIL.mil. 2013120901 3600 900 1209600 10800

mil. 1800 IN RR SIG SOA 8 1 21600 20131216182101 20131209182101 51921 mil. VtXu
YT/sn5p5dfjC1MH8raTXLXLm/LJ6EnEOCF5B3DZMm3UWKHkqG2X gP0NA7eZ13SB5SYoLdC+5nQETYOhX4m6h4LvMeTES08k4M05AgDIN
ov AlwZ3Cp3XSsV91jcC2uhr26JrR2BhabNv51Nz5thA38V5VbvdhG2H P4I0Anc=
UN6A0H9U9V Divine91LQ1KM82DKNPN6QHG.mil. 10800 IN NSEC3 1 0 10 EAAA UQ3DUCBQJQR7B1SFSE0J3ANEN2R04V4LL NS SOA RR SIGN DNSKEY NSEC3PARAM
UN6A0H9U9V Divine91LQ1KM82DKNPN6QHG.mil. 10800 IN RR SIG NSEC3 8 2 10800 20131216182101 20131209182101 51921 mil. lYfFhajnoMfwo+pcYfdbgKh5n/jg01w+h5faIoafAzOPVEjJrr5Mke A2p+SBgS9qCZ0kuIuoFSlvU/lxyG801337Es01G/USJUP
qCWPoKl98 BpFXqrNWIFZivSe3wYNbE2aK4CPzZhb8hqnM9u+nV05y6p0HkgHEpT RyK/woc=
LRIPTTJ7TQATH6L9LRPT94KL5K4A.mil. 10800 IN NSEC3 1 0 10 EAAA LSJHCK0HBN8GRT540MJ6PS9KD39BE5D NS
LRIPTTJ7TQATH6L9LRPT94KL5K4A.mil. 10800 IN RR SIG NSEC3 8 2 10800 20131216182101 20131209182101 51921 mil.

lmbtgVTXY5yWyxDF3M85OLOWYSERK9EITj9KtVMynR1EImGwMZalkd FVX9D+iXm/NNJahiZjD7CTdN9W88mNP0L7vWwVkbLj2U9qJKI
REpcVpQFk 3efnCdt/Gx8UvcymIHZe0CNMJ74FbM2DXHUCVbkw6YYIPR4M3sXclJaB 2FhKGsc=
QNI13VIVNF2E3K6GFA7K55Q8T08RQ9.mil. 1000 IN NSEC3 1 0 10 EAAA R5I410NG3VPRU3H05V9UB2NC0UN9HT1 NS DS RSS IG
QNI13VIVNF2E3K6GFA7K55Q8T08RQ9.mil. 1000 IN RR SIG NSEC3 8 2 10800 20131216182101 20131209182101 51921 mil.

LffNnMzthb1/ePjNhx188rk/krcjx3vUimIT6iCuAA7N4tZvG3nTrLXU6 a9TYL9JQ9S11zmdUpnfSsHjqx1RdN3w7tcFlnLfgphzkKo
EbLtt8tiq +fN5lJAFjHkwBGcllq8lwzBnDhSQC/QTZw+kLJD/27L79za74MXUj1Le GVijJS4=

;; Query time: 154 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Tue Dec 10 08:43:49 2013
;; MSG SIZE rcvd: 1034
Making a Validating Resolver

• See Proj 7x
Weaknesses of DNSSEC
Lack of Protection Between User Devices and Resolvers

- Attacker in the middle has enough info to perfectly forge responses
  - Unless DNSSEC is enforced in the client's browser
Lack of Protection of Glue Records

- DNSSEC only protects Resource Records if they are authoritative in the zone.
- Glue records are resource records that facilitate a resolution that is delegated from a parent zone to a child zone.
- They are owned by the child zone but appear in the parent zone.
- So they are non-authoritative.
- Not protected by DNSSEC.
Key Changes Don't Propagate

• Keys can
  – Roll over
  – Be Revoked
  – Signatures can expire

• These events do not propagate down the DNS tree
NSEC3 Denial of Service

• If a zone is insecurely delegated to another zone
• And it includes NSEC3 records
• A MITM can block resolution of a domain
• Or delegate the resolution to another server and change the A record
• Enabling spoofing, cookie-stealing, etc.
Re-Addressing Replay Attack

- If a server moves to a new hosting provider
- The old IP address can be used until the signatures expire
- Because there's no way to push signature expiration from authoritative servers to resolvers
NSEC3 Still Allows Zone Walking

• An offline dictionary attack can be used to guess the hashes
• It's the same as cracking password hashes
No Protection of DNS or Lower Layer Header Data

• The AD flag is part of the DNS header
  – Authenticated Data flag, indicating that the response data was verified by DNSSEC
  – Can't be trusted, because
• Can be changed by a MITM
• DNSSEC won't detect that
DNSSEC Data Inflate Zone Files and DNS Packet Sizes

- Small requests lead to large responses
- UDP allows spoofing the source IP address
DNSSEC Increases Computational Requirements on Validators and Servers

• Verifying signatures
• Calculating hashes for NSEC3 records
DNSCurve
Encrypted Packets

- Operates at layer 2 (Data Link)
- Uses Elliptic Curve Cryptography
- Each request and response packet is encrypted in its entirety
  - DNSSEC doesn't encrypt any data, it just signs it
- ECC keys are 256 bits long
- Calculation is much faster than RSA
DNSCurve Limitations

• Not yet an IETF standard
• Does not provide end-to-end security
  – Just from endpoint to resolver
• Name-server names need to be longer to include a Base-32 encoded 53-byte public key
  – Makes responses even larger
• DNS Queries may exceed 255 bytes, which will be dropped by old middleware
• Key compromise requires renaming the server
  – And manual update of NS record in the parent zone