DNSSEC Overview

ARIN+NANOG On The Road

Duane Wessels

Verisign Labs

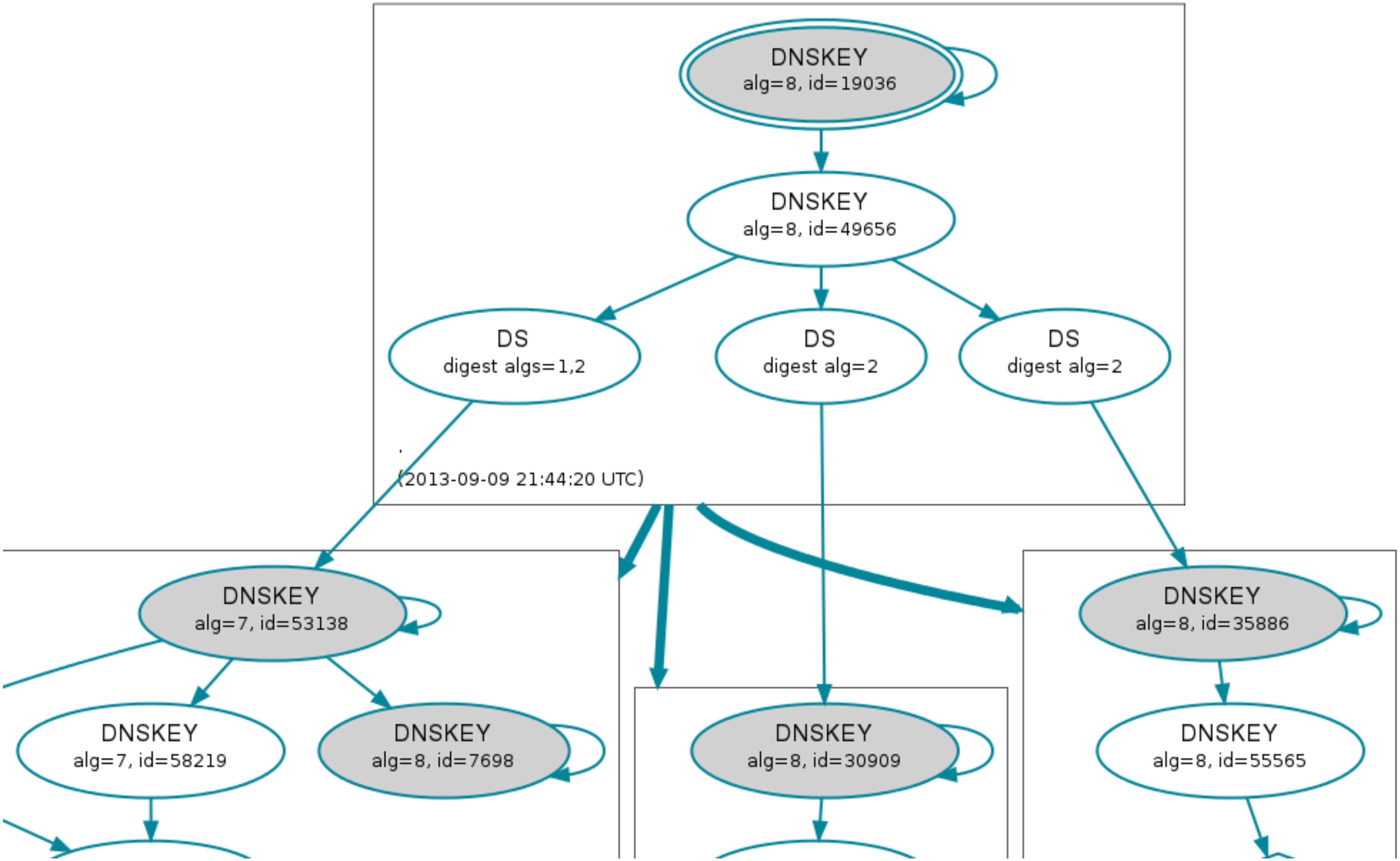
Version: 2012-02-05

VERISIGN™

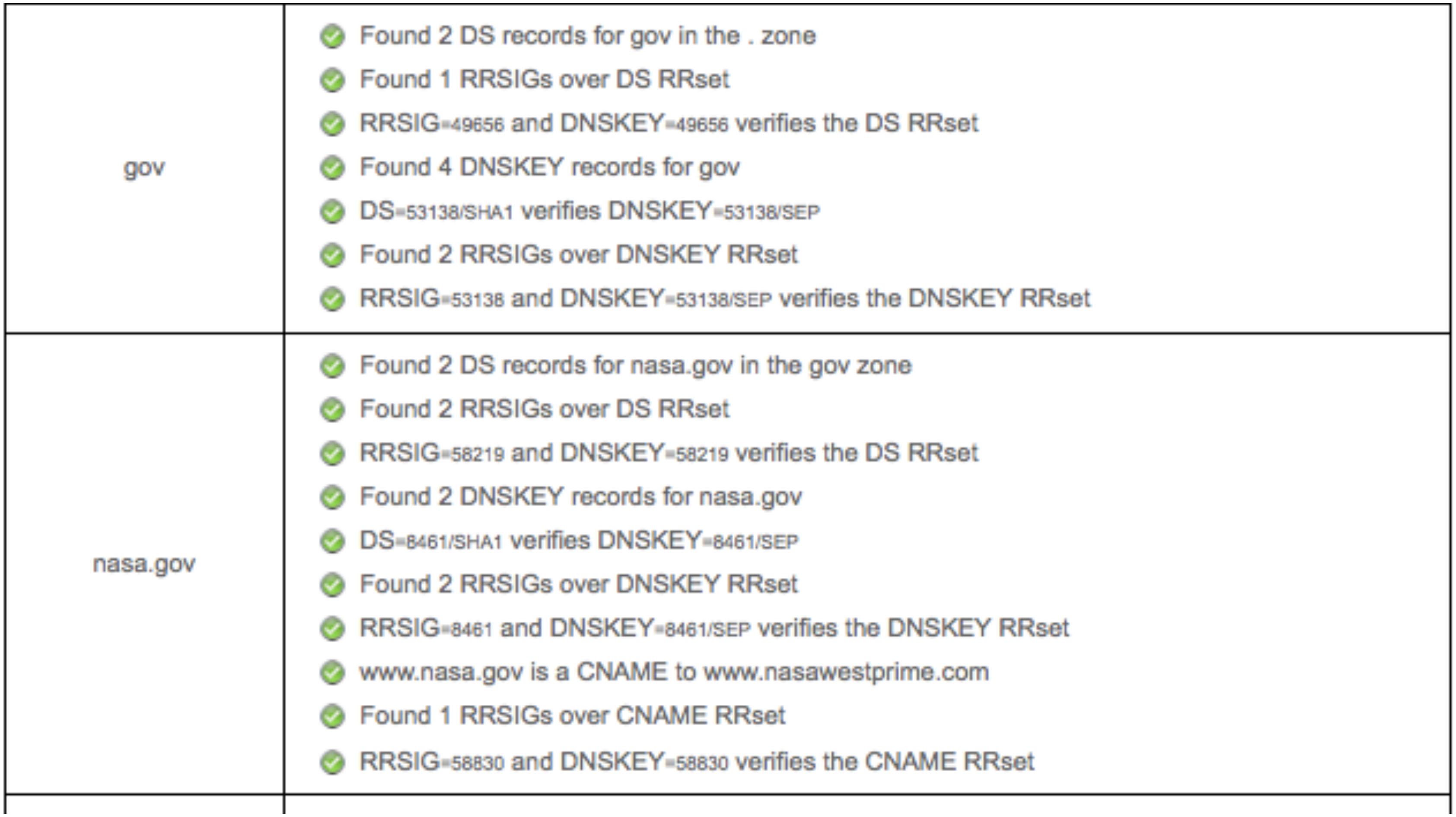
DNS Security

- DNS has no security
- One UDP packet for query, one UDP packet for response
- Must rely on source IP to match response to query
- Easily spoofed
- Clever resolvers help a lot
- But we need something better

Why do you need to know this stuff?



Why do you need to know this stuff?



gov	✓ Found 2 DS records for gov in the . zone ✓ Found 1 RRSIGs over DS RRset ✓ RRSIG=49656 and DNSKEY=49656 verifies the DS RRset ✓ Found 4 DNSKEY records for gov ✓ DS=53138/SHA1 verifies DNSKEY=53138/SEP ✓ Found 2 RRSIGs over DNSKEY RRset ✓ RRSIG=53138 and DNSKEY=53138/SEP verifies the DNSKEY RRset
nasa.gov	✓ Found 2 DS records for nasa.gov in the gov zone ✓ Found 2 RRSIGs over DS RRset ✓ RRSIG=58219 and DNSKEY=58219 verifies the DS RRset ✓ Found 2 DNSKEY records for nasa.gov ✓ DS=8461/SHA1 verifies DNSKEY=8461/SEP ✓ Found 2 RRSIGs over DNSKEY RRset ✓ RRSIG=8461 and DNSKEY=8461/SEP verifies the DNSKEY RRset ✓ www.nasa.gov is a CNAME to www.nasawestprime.com ✓ Found 1 RRSIGs over CNAME RRset ✓ RRSIG=58830 and DNSKEY=58830 verifies the CNAME RRset

DNSSEC Timeline

- 1993: Discussion of secure DNS begins
- 1994: First draft of possible standard published
- 1997: RFC 2065 published (DNSSEC is an IETF standard)
- 1999: RFC 2535 published (DNSSEC standard is revised)
- 2005: Total rewrite of standards published
  - RFC 4033 (Introduction and Requirements)
  - RFC 4034 (New Resource Records)
  - RFC 4035 (Protocol Changes)
- July 15, 2010: Root zone signed
- July 29, 2010: *.edu* signed
- December 9, 2010: *.net* signed
- March 31, 2011: *.com* signed

What DNSSEC Does

- DNSSEC uses digital signatures based on public key cryptography to provide:
  - Data origin authentication
    - “Did this DNS response really come from the *foo.com* zone?”
  - Data integrity
    - “Did an attacker (e.g., a man-in-the-middle) modify the data in this response since it was signed?”
- Bottom line: DNSSEC offers protection against spoofing of DNS data

What DNSSEC Doesn’t Do

- DNSSEC does not:
  - Provide any confidentiality for DNS data
    - I.e., no encryption
    - Most data in the DNS is public, after all
  - Address attacks against the name server itself
    - Denial of service,
    - Packets of death,
    - etc.

Key Pairs

- In DNSSEC, each zone has a public/private key pair
- The zone's public key is stored in the new **DNSKEY** record
- The zone's private key is kept safe
  - Private key storage options in increasing order of paranoia:
    - In a file readable only by root
    - In an encrypted file (decrypted only for signing)
    - Stored offline
    - In an HSM (Hardware Security Module)

The DNSKEY Record

```
test.com.        DNSKEY  256 3 5 (
                         AwEAAda013Wp4CQaUBrExCIRZCYpT5K93FIP
                         vOXfTkgT4LtMzEwRYnAONhKqpAaC7rAm2Jn+
                         VlYnzIqmwELmn0EqI/e7cV8Bao94dX3xdcK+
                         kZ6t5Of1hOLalyn/nsKZlH247VsEE62lHQNB
                         4nxPBHIpwURLqd9ilTsSeLxG56PdCVuJ
                         ) ; key id = 41148
                 DNSKEY  257 3 5 (
                         AwEAAckFh2HajtLkZr5JpNxjuhwnCOSlMuoV
                         ZKs+EfmrEoQ+oUs1KM5Nc93XPdq4WTbNwBi8
                         MYzdBDVZQys0byZzrm3VaPjJ/FIFOG8unhyn
                         mWUMmk4azYYvq0YOSbJf1vzAJbF842+a3hFm
                         5vTvuKZ8w9EhPd0rim0MBCV3jNetk/E9
                         ) ; key id = 46894
```

- DNSKEY record’s fields:
  - **256** or **257**, the 16-bit flags field
    - Bit 2^8 is set to indicate a DNSSEC zone key
    - Bit 2^0 is set to indicate a key-signing key (KSK)
  - **3**, the protocol octet
    - Will always be 3 to signify DNSSEC
  - **5**, the DNSKEY algorithm number (RSA with SHA1)
  - The public key itself, in base64
    - 1024-bit RSA keys in this example

Digital Signatures

- A zone's private key signs each resource record set (RRset) in a zone
  - RRset: records with same *owner*, *class* and *type*
    - Domain name *www.test.com*, class IN, type A
    - *www.test.com* / IN / A
- Each RRset's digital signature is stored in an **RRSIG** record
- Not all information in a zone is signed:
  - Delegation information is not signed
    - Delegating NS records and corresponding A and AAAA records (glue)
  - Authoritative copies of these records in the child zone, not the parent

The RRSIG Record

```
www.test.com.           86400   A       192.0.2.1
                        86400   A       192.0.2.2
                        86400   RRSIG   A 5 3 86400 20090507235959 (
                                        20090501000000 41148 test.com.
                                        s8dMOWQjoTKEo1bsK+EYUY+32Bd84300FcJf
                                        lqthv1u60DVDVobllhqt0AaiD/dlnn7Yask6
                                        xGe0u0lBbm06bsq28KP5rf9cR4bmmx68V1pQ
                                        IKcm1Tx/Y1ixJHFiRMxMoEoiZp1sR9x/YIHL
                                        C7F+4Xuk8sePEzz9vA92puhtkSA= )
```

- RRSIG record’s fields:
  - **A**, the type of records signed
  - **5**, the digital signature algorithm used (RSA with SHA1)
  - **3**, the number of labels in the signed name
  - **86400**, the original time-to-live on the records signed
  - **20090507235959**, when the signature expires
  - **20090501000000**, when the records were signed
  - **41148**, the key ID/tag/footprint
  - **test.com.**, the signer’s name
  - Finally, the digital signature itself, in base64

Proving Something Doesn’t Exist

- Negative errors:
  - Name Error (NXDOMAIN)
  - “No such data” (NOERROR/0)
- How do you prove cryptographically that the RRset doesn’t exist?
- Could sign negative responses “on the fly”
- Or sign something ahead of time: the **NSEC** record

The NSEC Record

- NSEC record used to prove non-existence of DNS data
- The NSEC record…
  - Resides at a given domain name
  - Specifies what types exist at that name
  - Points to the next domain name in the zone
- The NSEC record spans a gap between two domain names in a zone
- Notion of a “next” record implies a canonical order
- Labels in a domain name are sorted by:
  - Shifting all characters to lowercase
  - Sorting non-existent bytes ahead of “0”
  - Sorting lexicographically from the highest-level label to the lowest

Ordering a Zone

- So the following example zone:

```
test.com.     SOA         ns.test.com.     root.test.com. (
                             2009041800 1h 10m 30d 1d )
              NS          ns.test.com.
              A           10.0.0.1
              MX          0 mail.test.com.
ns            A           10.0.0.1
mail          A           10.0.0.2
www           A           10.0.0.3
ftp           CNAME       www.test.com.
west          NS          ns.west.test.com.
ns.west       A           10.0.0.5
east          NS          ns.east.test.com.
ns.east       A           10.0.0.6
```

Ordering a Zone

- Would sort to:

```
test.com.                 SOA      ns.test.com.     root.test.com. (
                                      2009041800 1h 10m 30d 1d )
test.com.                 NS       ns.test.com.
test.com.                 A        10.0.0.1
test.com.                 MX       0 mail.test.com.
east.test.com.            NS       ns.east.test.com.
ns.east.test.com.         A        10.0.0.6
ftp.test.com.             CNAME    www.test.com.
mail.test.com.            A        10.0.0.2
ns.test.com.              A        10.0.0.1
west.test.com.            NS       ns.west.test.com.
ns.west.test.com.         A        10.0.0.5
www.test.com.             A        10.0.0.3
```

Adding NSEC Records

- And here’s the zone with NSEC records added:

```
test.com.                  SOA      ns.test.com. root.test.com. (
                              2009041800 1h 10m 30d 1d )
test.com.                  NS       ns.test.com.
test.com.                  A        10.0.0.1
test.com.                  MX       0 mail.test.com.
test.com.                  NSEC     east.test.com. A NS SOA MX NSEC
east.test.com.             NS       ns.east.test.com.
east.test.com.             NSEC     ns.east.test.com. NS NSEC
ns.east.test.com.          A        10.0.0.6
ns.east.test.com.          NSEC     ftp.test.com. A NSEC
ftp.test.com.              CNAME    www.test.com.
ftp.test.com.              NSEC     mail.test.com. CNAME NSEC
mail.test.com.             A        10.0.0.2
mail.test.com.             NSEC     ns.test.com. A NSEC
ns.test.com.               A        10.0.0.1
ns.test.com.               NSEC     west.test.com. A NSEC
west.test.com.             NS       ns.west.test.com.
west.test.com.             NSEC     ns.west.test.com. NS NSEC
ns.west.test.com.          A        10.0.0.5
ns.west.test.com.          NSEC     www.test.com. A NSEC
www.test.com.              A        10.0.0.3
www.test.com.              NSEC     test.com. A NSEC
```

Notes on NSEC

- The final NSEC “wraps around” from the last name in the ordered zone to the first
- Each NSEC record has a corresponding RRSIG

NSEC In Use

- Looking up *north.test.com*: the name doesn't exist
  - The response has return code NXDOMAIN and includes:

```
mail.test.com.  NSEC  ns.test.com. A NSEC
```

“No domain names in the zone between *mail.test.com* and *ns.test.com*”

- Looking up TXT records for *mail.test.com*: the name exists but has no TXT records
  - The response has return code NOERROR, no records in the answer section, and includes:

```
mail.test.com.  NSEC  ns.test.com. A NSEC
```

“No TXT records for *mail.test.com*, only A and NSEC”

Chain of Trust

- There are no certificates in DNSSEC
- The trust model is rigid
- Only a zone’s parent can vouch for its keys’ identity
- The “chain of trust” flows from parent zone to child zone

Types of Keys

- Signed zone has DNSKEY RRset at apex
  - Usually contains multiple keys
  - One or more **key-signing keys (KSKs)**
  - One or more **zone-signing keys (ZSKs)**
- KSK
  - Signs only the DNSKEY RRset
- ZSK
  - Signs the rest of the zone

- Why two types of keys?
  - KSK change requires interaction with parent
  - ZSK change has no external dependencies

Delegation Signer (DS) Records

- The **Delegation Signer (DS)** record specifies a child zone’s key (usually the KSK)
  - DS record contains a cryptographic hash of child’s KSK
- A zone’s DS records only appear in its parent zone
  - Along with NS records at a delegation point
- DS records are signed by the parent zone

The DS Record

```
; This is an excerpt of the .com zone data file
test.com.            86400   NS      ns1.test.com.
                     86400   NS      ns2.test.com.
                     86400   DS      46894 5 1 (
                                     A6879FC55299A0985CF0D72B0EDAD528C10E
                                     FD00 )
                     86400   DS      46894 5 2 (
                                     BEA484A06FBB93034A3FD9CE8C7F37391B0B
                                     FAA2AA58B1EB09A5B59DFBAF304B )
                     86400   RRSIG   DS 5 2 86400 20090507235959 (
                                     20090501000000 810 com.
                                     D05vBDjM9hb01uaMk/GYG81aZWGCDp/Hn90P
                                     vpthFK4gPMwCvX+r3HQeKyWYzbEnr/mIAO1L
                                     6OLhi5vvbD48+UulDyplXVJ37nJrt9DiFN75
                                     z7nk2rjEctoNSZ3BI1NVwtvFl5zBHSDqih2x
                                     /dRJQ2ICfDVIdC3tdV8IPV0zJWE= )
```

- DS record's fields:
  - **46894**, the key ID/tag/footprint
  - **5**, the DNSKEY algorithm number (RSA with SHA1)
  - The digest type: **1** is SHA-1, **2** is SHA-256
  - Finally, the digest, in hexadecimal

Unsigned Zone Example: *example.com*

```
example.com.              SOA      <SOA stuff>
example.com.              NS       ns1.secure-hoster.net.
example.com.              NS       ns2.secure-hoster.net.
example.com.              A        192.45.56.67
example.com.              MX       10 mail.example.com.
mail.example.com.         A        192.45.56.68
www.example.com.          A        192.45.56.67
```

Signed Zone Example: *example.com*

```
example.com.                SOA      <SOA stuff>
example.com.                RRSIG    SOA <RRSIG stuff>
example.com.                NS       ns1.secure-hoster.net.
example.com.                NS       ns2.secure-hoster.net.
example.com.                RRSIG    NS <RRSIG stuff>
example.com.                A        192.45.56.67
example.com.                RRSIG    A <RRSIG stuff>
example.com.                MX       10 mail.example.com.
example.com.                RRSIG    MX <RRSIG stuff>
example.com.                DNSKEY   <Key that signs example.com DNSKEY RRSet>    ; KSK
example.com.                DNSKEY   <Key that signs the rest of example.com zone> ; ZSK
example.com.                RRSIG    DNSKEY <RRSIG stuff>
example.com.                NSEC     mail.example.com. SOA NS A MX DNSKEY RRSIG NSEC
example.com.                RRSIG    NSEC <RRSIG stuff>
mail.example.com.           A        192.45.56.68
mail.example.com.           RRSIG    A <RRSIG stuff>
mail.example.com.           NSEC     www.example.com. A RRSIG NSEC
mail.example.com.           RRSIG    NSEC <RRSIG stuff>
www.example.com.            A        192.45.56.67
www.example.com.            RRSIG    A <RRSIG stuff>
www.example.com.            NSEC     example.com. A RRSIG NSEC
www.example.com.            RRSIG    NSEC <RRSIG stuff>
```

Trust Anchors

- You have to trust somebody
- DNSSEC validators need a list of trust anchors
  - Keys (usually KSKs) that are implicitly trusted
  - Analogous to list of certificate authorities (CAs) in web browsers
- Trust anchor store can be updated via:
  - Manual process
    - Static configuration
  - DNSSEC “in band” update protocol
    - RFC 5011
  - Other trusted update mechanism
    - From name server or operating system vendor

Example Chain of Trust

- We are validating A records for *www.verisign.com*.

- Trust anchor for root zone **KSK ➔**
  - Statically configured in the DNSSEC validator
- root **KSK ➔** root **ZSK ➔** *.com* **DS ➔**
  - In the root zone
- *.com* **KSK ➔** *.com* **ZSK ➔** *verisign.com* **DS ➔**
  - In the *.com* zone
- *verisign.com* **KSK ➔** *verisign.com* **ZSK ➔** *www.verisign.com* **A**
  - In the *verisign.com* zone

NSEC3

- **NSEC3** is an alternative to NSEC providing:
  - Non-enumerability
  - Opt-Out
- Significant standards effort by Verisign, Nominet (*.uk* registry) and DENIC (*.de* registry)
- RFC 5155
  - Published February, 2008

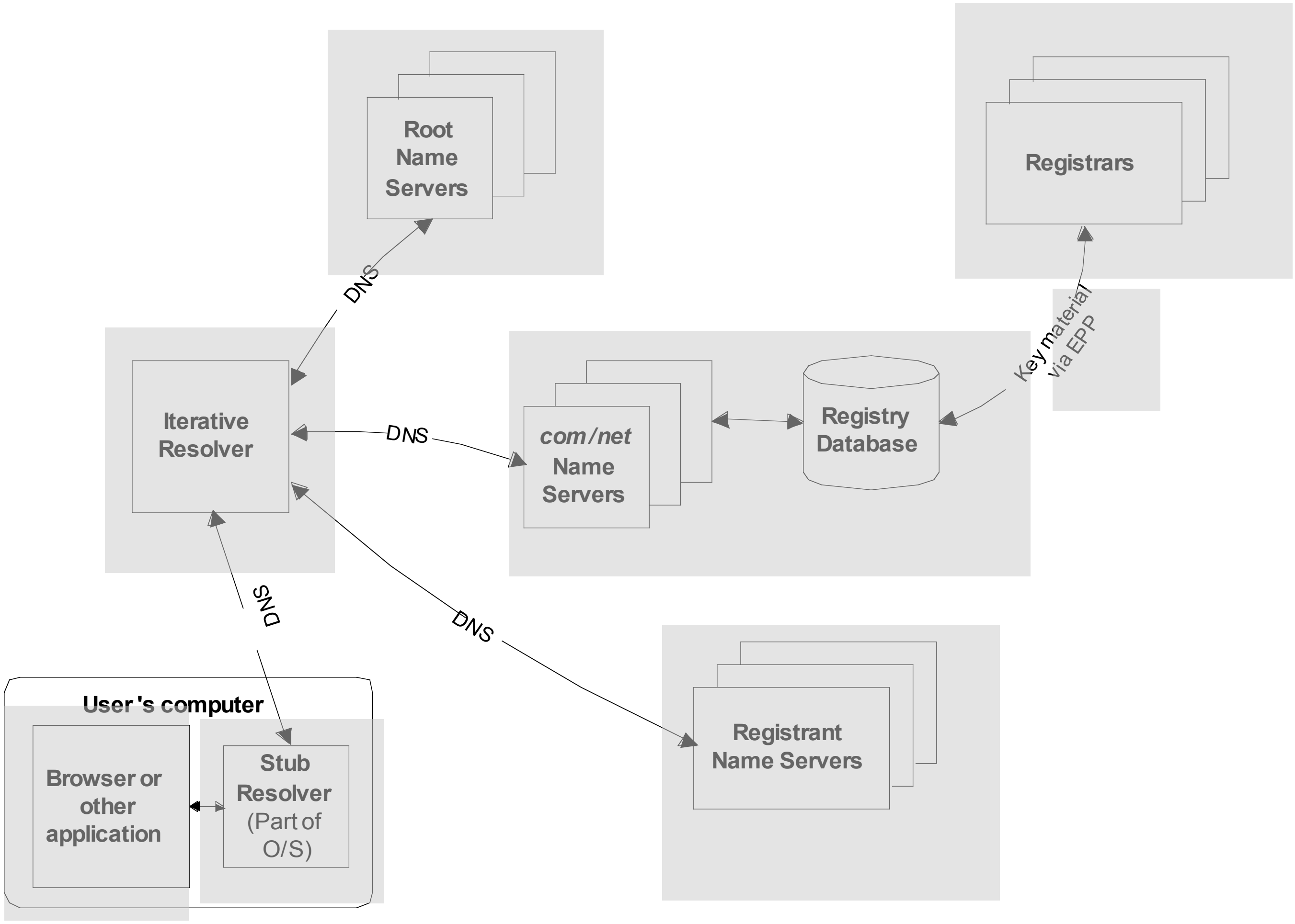
Non-Enumerability

- Stops zone enumeration via “zone walking” the NSEC chain
- NSEC3 chain is hash of names
- Example:
  - Zone: *alpha.com, bravo.com, charlie.com*
  - NSEC chain:
    - *alpha.com* ➔ *bravo.com* ➔ *charlie.com*
  - NSEC3:
    - H(*bravo.com*) ➔ H(*alpha.com*) ➔ H(*charlie.com*)
    - *adfjkhjim.com* ➔ *djadfjhifj.com* ➔ *qsfiudfiud.com*

Opt-Out

- Standard DNSSEC:
  - Every name in a zone has an NSEC
    - Including delegations (NS records)
- Opt-Out DNSSEC:
  - Only secure delegations have an NSEC
    - I.e., delegations to zones that are themselves signed
- Better for large zones like .com
  - Many names, but few secure delegations
  - Shorter NSEC3 chain
  - Fewer signatures
  - Smaller signed zone

Changes for DNSSEC



What will DNSSEC be used for?

- Protecting applications against DNS spoofing attacks
  - Recursive name servers will perform DNSSEC validation and throw away bad data before it reaches downstream clients
  - Eventually some stub resolvers and even applications may do their own DNSSEC validation
- Opening up DNS as a secure repository for various kinds of data
  - Web site authentication and privacy
    - X.509 certificates authenticated by DNSSEC
      - Self-signed or “stapled” to a particular Certificate Authority
    - IETF DANE Working Group
  - Mail origin authentication
  - SSH host key authentication
  - Publication mechanism for other public keys?
  - Secure routing information repository?

Questions?