DNSSEC

Joakim Hovlandsvåg

April 18, 2012

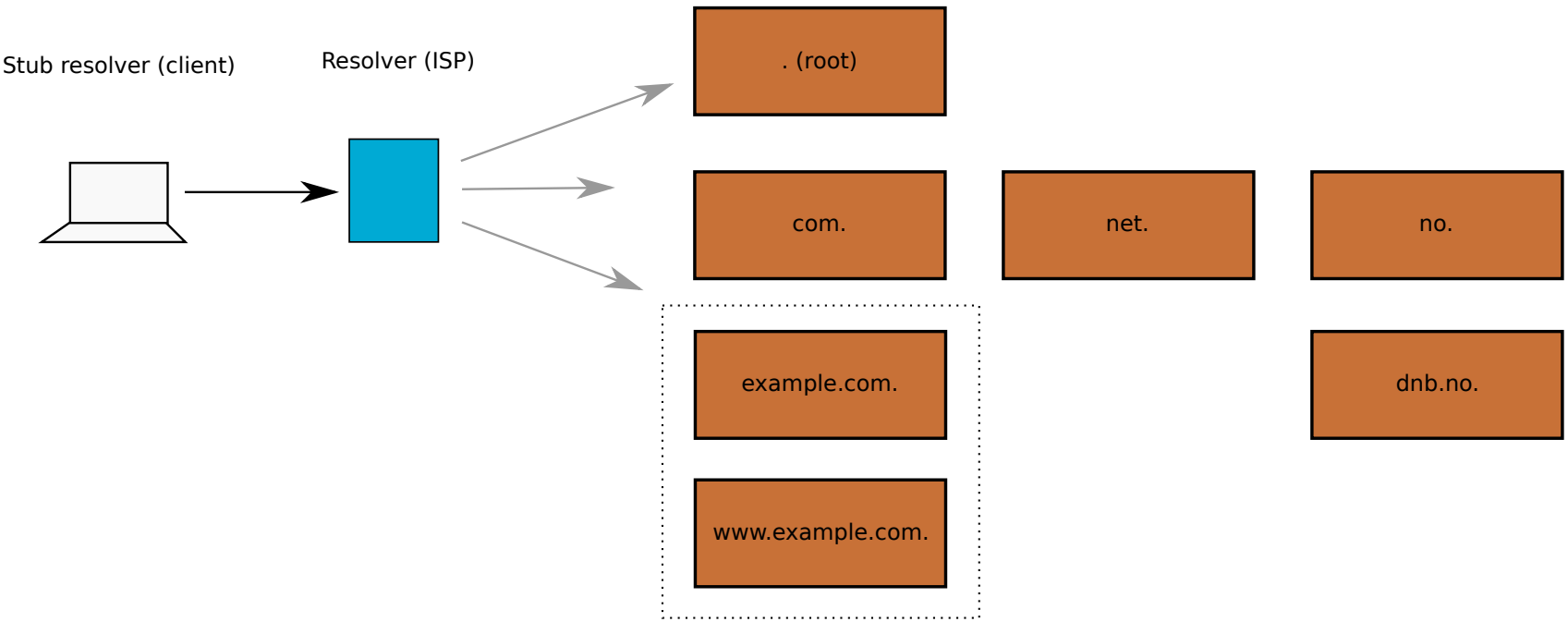
Outline

- DNS
  - History
  - Overview
  - Structure
  - Examples
- DNS vulnerabilities
- DNSSEC
  - Overview
  - Root certificate

History

- ARPANET started mapping addresses to names
- Before 1983: *HOSTS.TXT*, hosted at Stanford Research Institute
- 1983: DNS invented (RFC82[2-3])
- 1985: BIND
- a lot of superseding RFCs
- usage extended
- 1999: failed attempt for DNSSEC
- 2005: DNSSEC
- 2011: DNS root signed

DNS from a normal user's perspective



DNS query

- User wants *www.dnb.no*
- Browser needs its IP address and asks resolver (*/etc/resolv.conf*):

```
www.dnb.no.                     IN      A
```

- Resolver finds the answer and responds:

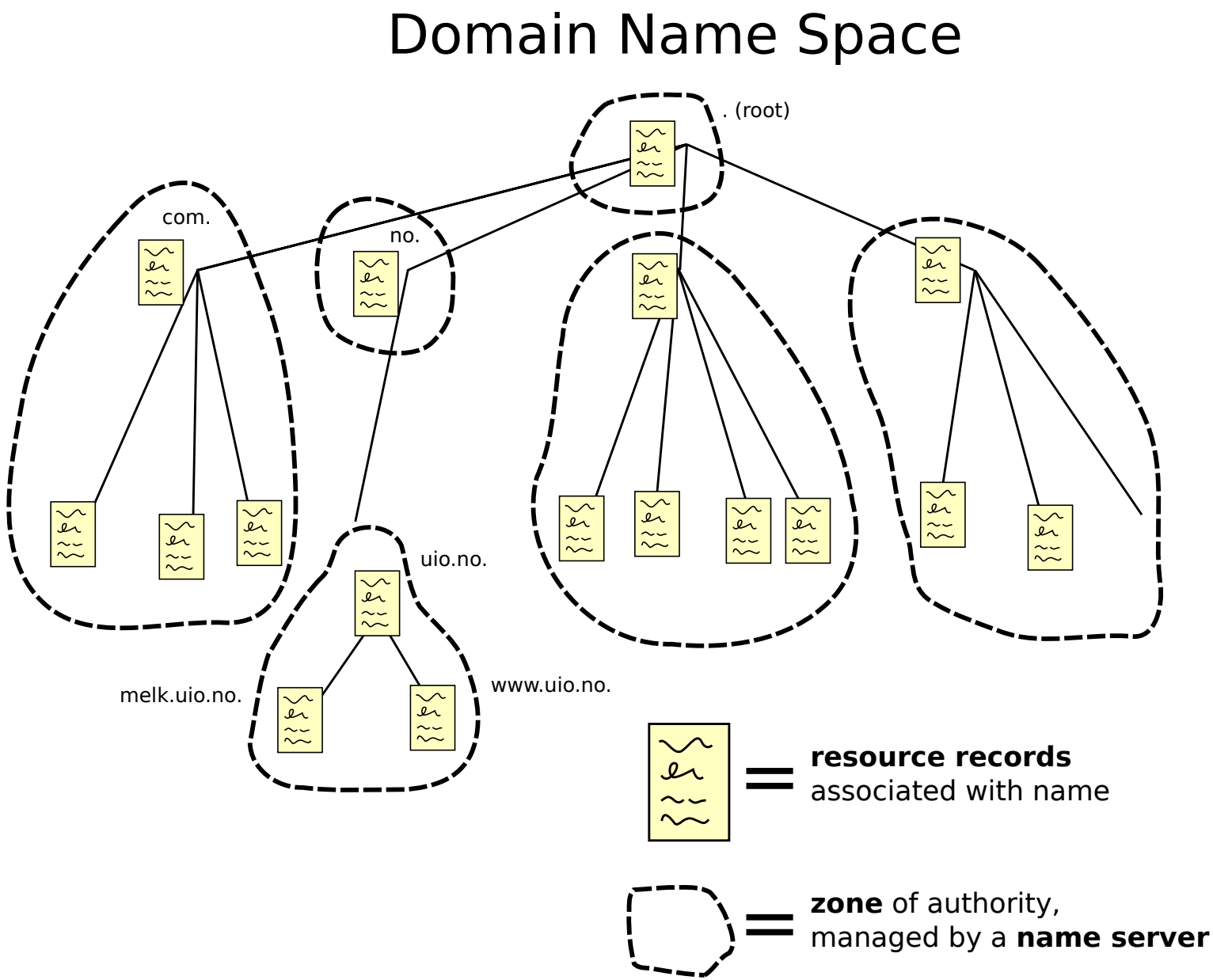
```
www.dnb.no.             243     IN      A       193.71.229.12
```

DNS content

- **A** - IPv4 addresses
- **AAAA** - IPv6 addresses
- **NS** - Authoritative Name Server, which should know more about the requested info
- **CNAME** - Canonical Name, aliases (e.g. *www.google.com*)
- **MX** - Address of mail server for given domain.
- **SRV** - Address of different servers at the given domain.
- **TXT** - Arbitrary text.

. . . and many more.

Structure of DNS



The root node

- 13 root nodes
- Run by different organizations.
- Authoritative data is actually off line, no root node is the authoritative one.

Structure of root nodes

13 root nodes.

```
.                       518400  IN      NS      a.root-servers.net.
.                       518400  IN      NS      b.root-servers.net.
.                       518400  IN      NS      c.root-servers.net.
.                       518400  IN      NS      d.root-servers.net.
.                       518400  IN      NS      e.root-servers.net.
.                       518400  IN      NS      f.root-servers.net.
.                       518400  IN      NS      g.root-servers.net.
.                       518400  IN      NS      h.root-servers.net.
.                       518400  IN      NS      i.root-servers.net.
.                       518400  IN      NS      j.root-servers.net.
.                       518400  IN      NS      k.root-servers.net.
.                       518400  IN      NS      l.root-servers.net.
.                       518400  IN      NS      m.root-servers.net.
```

Actually many servers behind these - anycast.

Map of root nodes

http://www.root-servers.org/

Example of DNS data

`dig`

DNS vulnerabilities

- DNS Cache poisoning - the Daminsky attack
- DNS Hijacking - give resolvers fake DNS responses
- Man-in-the-middle attacks

DNSSEC

- **Authentication** through signed public keys
- **Integrity** as all data is signed
- **Denial of existence** through a new RR
- Creating a **chain of trust** from the root node and down.

DNSSEC Overview

New Resource Records: DNSKEY, RRSIG and DS

DNSSEC elements - DNSKEY

- A zone's **public key**.
- Specifies
  - *Flags*, e.g. if it's a zone and/or a public key.
  - *Algorithm* of the key, e.g. DSA/SHA-1, RSA/SHA-1 or Elliptic curve.
  - The *public key* itself.

DNSSEC elements - DS

- **Delegation Signer** - parent zone's signature of current zone's DNSKEY.
- Specifies
  - *Algorithm* - e.g. DSA/SHA-1, RSA/SHA-1 or Elliptic curve.
  - *Digest type* - SHA-1.
  - A *digest* of the DNSKEY: `sha1(dnskey owner name | dnskey)`

DNSSEC elements - DS

- Why DS? Why not just let the parent node hold the childs' DNSKEYs?
- 22 millions zones under *com.* - how to sign these?
- What if *com.* updated its DNSKEY?

DNSSEC elements - RRSIG

- **Resource Record Signature** - signatures returned together with DNS responses.
- Specifies:
  - *Type of RR* that is covered, e.g. an *A* record.
  - *Algorithm* - e.g. DSA/SHA-1, RSA/SHA-1 or Elliptic curve.
  - *Signature expiration and inception time*
  - *Signer's name* - e.g. *example.com*.
  - The *signature*, constructed by:
    `sign(private key, RRSIG data | RR1 | RR2 ...)`

DNSSEC authentication

- Must already have the root key.
- Note: stub resolvers must trust other parties.
- BIND/dig: */etc/trusted-key.key* for the root's key.
- *dig +dnssec +sigchase...*

The key signing process

The single trust anchor!
The root was generated July 16th, 2010.
http://www.youtube.com/watch?v=b9j-sfP9GUU
http://data.iana.org/ksk-ceremony/

Other usage

Since:

- All data is authenticated
- DNS is easy to retrieve data from
- One could store arbitrary data

it makes perfect sense for:

- Storing SSH server's public keys.
- Storing GPG public keys for e-mail use. E.g. all employees in a company.
- Storing web server certificates, e.g. instead of X.509

Other usage

Since:

- All data is authenticated
- DNS is easy to retrieve data from
- One could store arbitrary data

it makes perfect sense for:

- Storing SSH server's public keys.
- Storing GPG public keys for e-mail use. E.g. all employees in a company.
- Storing web server certificates, e.g. instead of X.509