OpenDNSSEC training

Opening

Agenda

Time: Day 1: 10:00 – 17:00, Day 2: 09:00 – 16:00

- Introduction to DNSSEC and the OpenDNSSEC application
- Prerequisites for running OpenDNSSEC and description of the lab environment
- Hardware Security Modules
- SoftHSM installation and initialization
- Configuration files for ODS, conf.xml , kasp.xml and zonelist.xml
- Running OpenDNSSEC
- Testing
- Integration
- Monitoring
- Recovery planning
- Operational practices

Introduction

- Who am I?

Introduction

- Who are you?
- Any experience with DNSSEC?
- What are your expectations?

Goals

- Understanding of DNSSEC
- OpenDNSSEC
  - Install
  - Configure
  - Sign zones
- Integrate with your environment
- Basic troubleshooting

Lab environment

- Amazon Elastic Compute Cloud (EC2)
- One teacher server running *odslab.se*
- Two servers per group
  - Resolver – *resolverX.odslab.se*
  - Name server – *nsX.odslab.se*
- One domain per group
  - *groupX.odslab.se*

The lab

- Handouts with lab instructions
- Most of the labs are introduced by a presentation
- Group numbers and login credentials are handed out by the teacher

OpenDNSSEC training

Uploading the DS RR

Head start

- We need to create a chain-of-trust to our test domain, *odslab.se.*
- .SE distributes its zone every second hour (but are allowed to take up to five days).
- Need to this so that you can validate your subzones later in this lab.

Uploading the DS RR

- Creates a chain-of-trust.
- You do this when your zone is signed.
- We have prepared *odslab.se*, it is already signed.
- How you upload the DS RR depends on your registrar.
- Various APIs and web interfaces, some does not even support DNSSEC.

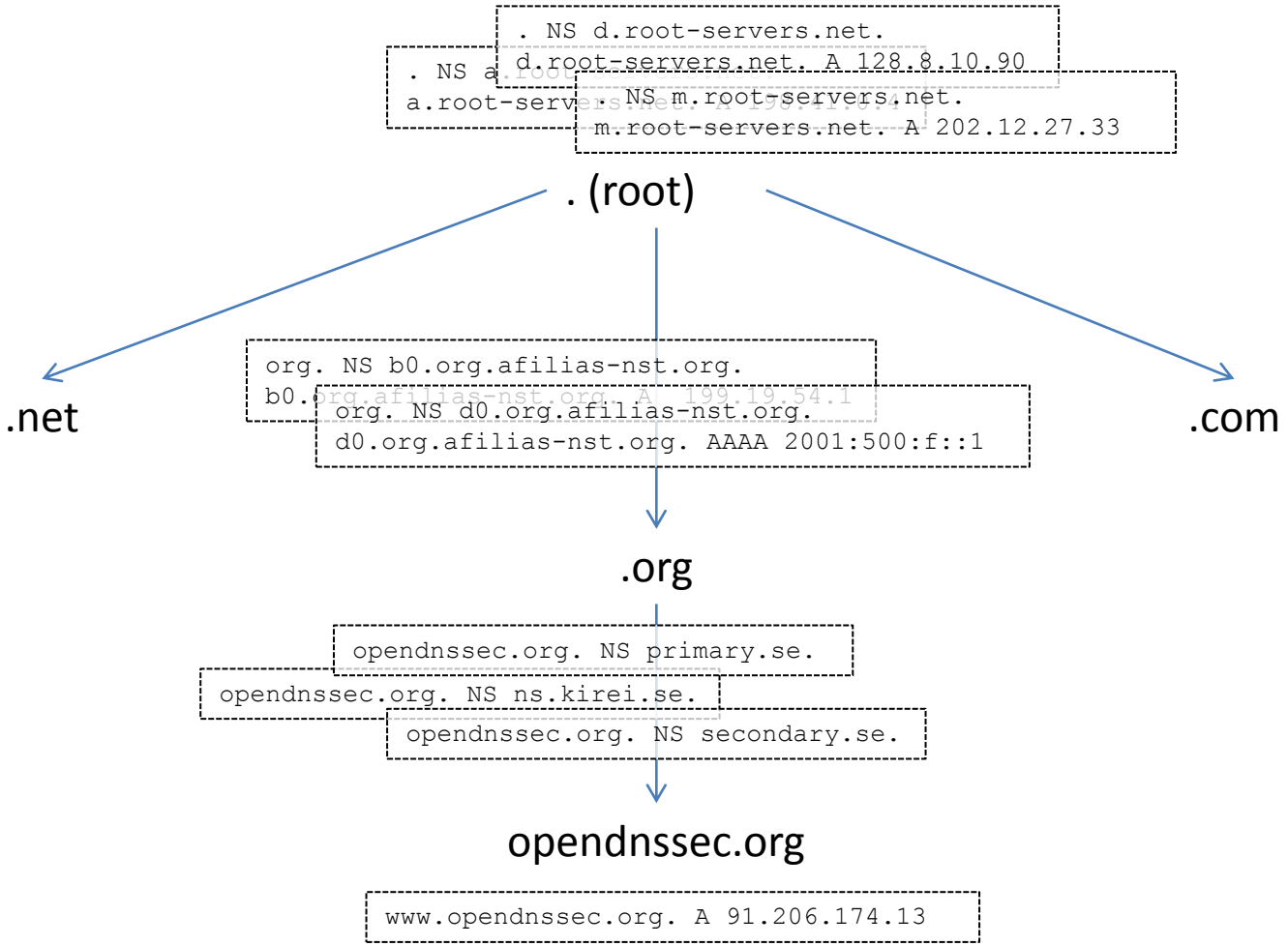
Live demo

- *odslab.se* is using the registrar SE Direkt.
- https://domanhanteraren.iis.se/lang/?set=en

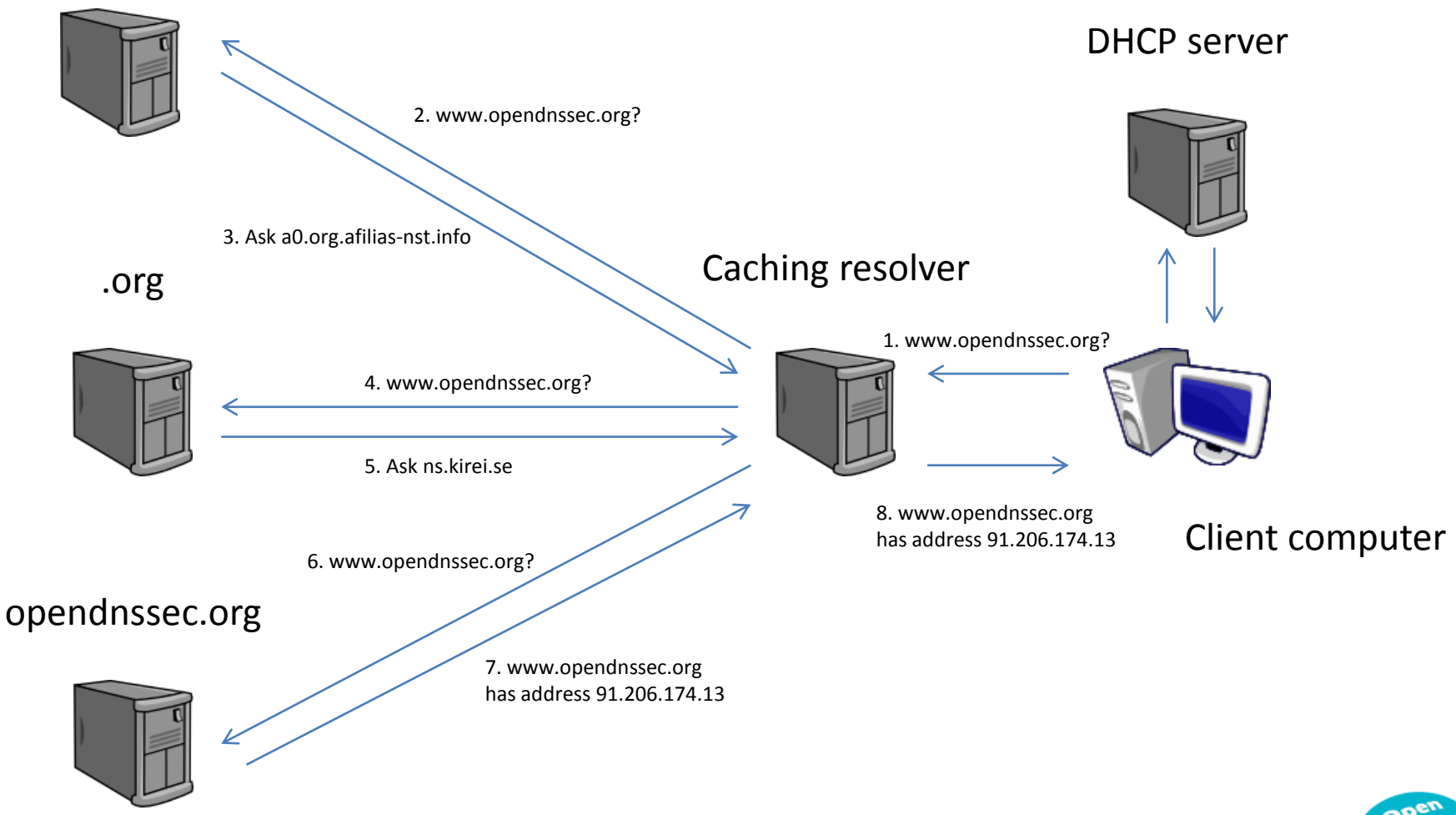
OpenDNSSEC training

DNSSEC introduction

The DNS Hierarchy



Resolving DNS



Vulnerabilities

- You cannot trust the DNS answer
- Various categories of threats
  - Denial of Service
  - Data integrity
    - Protocol issues – Cache poisoning, Query prediction
    - System corruption
    - Repository corruption
  - Privacy
    - Cache snooping
    - NSEC walk

What is DNSSEC?

- Domain Name System Security Extension
- An extension that is placed on top of DNS
- It gives:
  - Data Origin Authentication
  - Data Integrity
  - Denial of Existence
- By using digital signatures
- Fixes some of the protocol issues

Add crypto to the mixture

- Asymmetric crypto:
  - Asymmetric key pairs have a public and private key
  - Protect the private keys
  - Publish the public keys
- KSK:
  - The Key Signing Key - what you trust
  - Signs the Zone Signing Keys, ZSK
- ZSK:
  - The Zone Signing Key
  - Creates signatures of records in the zone - RRSIG

DNSKEY and RRSIG

```
opendnssec.org.      IN      DNSKEY  256 3 5
BQEAAAAB2WMDxqWR7cCadFXQmmR3jhfHekKf5uhUVxBFyzHGyHc1vVi0
u4w3Z+/96anmn+oTzuxGmYOPm3j+3AfatV3USD8b4DdkM35aNZ2iMyXd
lMFb+OgPUODl71nnxp2KGFu8oWtILLJMOAo5giitUpMFWmGAKJH/BbWh
WydF1fKwLuk=
opendnssec.org.      IN      DNSKEY  257 3 5
BEAAAAOhdFlVHeivG77Zos6htgLyIkBOn18ujX4Q7Xs6U7SDQdi6FBE5
OQ8754ppfuF3Lg1ywNLHQ5bjibquSG7TuCT6DWL3kw+hESYmWTeEev9K
RnxqTA+FVIfhJaPjMh7y+AsX39b8KVQ32IYdttOiz30sMhHHPBvL4dLC
4eCQXwUbinHRWSnKpKDXwuaUUtQkPqkEc4rEy/cZ3ld408vMlcc73OcK
t+ttJeyQR1dJ0LoYHvH0WBzIWg3jUPmz/hSWrZ+V2n0TISQz0qdVGzhJ
vahGvRstNk4pWG1MjwVgCvnc18+QiEV4leVU7B4XjM9dRpIMzJvLaq+B
d8CxiWvjpSu/

opendnssec.org.     IN      RRSIG   DNSKEY 5 2 3600 20110705003007 20110625003007
40957 opendnssec.org. PXW2Zj3HM2annBMGGHormcyIUZF4s+KZIKynoNfSyqHmiTghUDxVUStF
tzp88ZlHLV+0CYQU4zY20RI9kGg7Iwc+jF8BGjoJfIrNtt6ado9sBrqD
znK/fal6fsFl7HuhRke68P5mwQETKOTV3S0Tcfz6krmqofbTAq5qwkqf
CBX4Wm6csZHWVF+pUlNhPumJpbnI6mHNcRvVSx07D3TGRT4ZF/1s38md
GaFkSKc2zxgGCOSyWfUml93AQ5Zox+l1hfGr3NZd7MAynklwZSrY/JzK
mUN24n2wmjrNNFaQuXbjO2T+Mqm2PB3yweYxyh2kKryf5Oc3tkglr1jP
zsuoWg==
opendnssec.org.     IN      RRSIG   DNSKEY 5 2 3600 20110705003007 20110625003007
49829 opendnssec.org. BlZZUWXTpQ8Ur0MBJxgHASarKfWREOTABaW+d/zIaFtUjhicQUjm2IUx
4O84gxslKvk/uhwfm0qYII+Mlz3IX93e6Ml8EC+O/0zFPEXjwQRmHplC
+qjyOAONHOfyqG0El+da33tr+E+VBtigTN5GyqSDfZ/zuPRkiYr8Uxzg
CJ0=
```

Signatures?

- A signature is an encrypted hash of data.
- The key used for encryption is the private key, and the signature can be verified by decrypting the hash with the public key.
- A hash is a checksum of a set of data. Typical checksum algorithms are MD5, SHA-1 and SHA-256. MD5 is considered vulnerable.

DNSKEY algorithms

- Different DNSSEC algorithms:
  - RSAMD5
  - DSA
  - RSASHA1
  - DSA-NSEC3-SHA1
  - RSASHA1-NSEC3-SHA1
  - RSASHA256
  - RSASHA512
  - ECC-GOST

NSEC

- Proof of non-existence
- You want to protect anybody from performing a DoS attack against a name in DNS. That is done with NSEC.

```
mail.opendnssec.org.  IN  NSEC  svn.opendnssec.org. CNAME RRSIG NSEC
```

NSEC3

- NSEC makes zone walking possible
- Uses the hash of the domain name
- Requires more resources from resolver and the name server.

```
7oreb1sb9elhfqfp53bqqde6bcdm5eo3.groupx.odslab.se.  IN  NSEC3 1 0 5
3A5BF749D1330DE3OTANAROMKJB00QC2G6K2IT2GU2SB4DOA CNAME RRSIG
```

Zone file without DNSSEC

```
$TTL 60
@       IN SOA nsX.odslab.se. test.odslab.se. (
                    2011062100 ; serial
                    360        ; refresh (6 minutes)
                    360        ; retry (6 minutes)
                    1800       ; expire (30 minutes)
                    60         ; minimum (1 minute)
                )
        IN NS nsX.odslab.se.
www     IN CNAME nsX.odslab.se.
```

Zone file with DNSSEC

```
groupX.odslab.se.       60 IN SOA nsx.odslab.se. test.odslab.se. (
                                2011062145 ; serial
                                360        ; refresh (6 minutes)
                                360        ; retry (6 minutes)
                                1800       ; expire (30 minutes)
                                60         ; minimum (1 minute)
                                )
groupX.odslab.se.       60 IN RRSIG SOA 8 3 60 20110628103724 (
                                20110628083552 44494 groupx.odslab.se.
                                NJ5lIdcdw3TJlSjTd5W/Gk1CtgZu2VfXAVIF49em/jdm
                                pA1JnejkwPAfb0TjdcXBUH6cQ2XIHobjgEJEpWRM9G/W
                                W7DYJZmdo6o09YrMexTLCZLcq6eyjTpS8TmwmconuNEN
                                FiCkBztqgHlyw0Teg9sw/1E0UVwGKKgd0SOv8Nw= )
groupX.odslab.se.       60 IN NS nsx.odslab.se.
groupX.odslab.se.       60 IN RRSIG NS 8 3 60 20110628103609 (
                                20110628083552 44494 groupx.odslab.se.
                                K3Yxcz25nv0m8SZDHkh0YXPBrZ0+78hVsT7FD4A9GZ9m
                                3sHpkpfzjZ/Bee+lgwZZGIJKmMfyRtQQon7oCa2Z9xe9
                                L/D9KQzPzZbZCMrOxG/usSZ+LhwYuN3b0Kl2BIhklji5
                                fBN6aEsyhw+hiV9ibobzqKe5bMnxaa9IfMscV1c= )
groupX.odslab.se.       120 IN DNSKEY 256 3 8 (
                                AwEAAasv0uyeTp5kIaw/fwPyQncY06YMn370lczC5SCx
                                veUNQXLhihm+tV/lTvkWd5GHg/ebjTPSR6mqB/jTu7CH
                                /iNhprxdnh3lVW7FjFpC5tDfFiHyDM97q8A+4lnBmiB4
                                SZJR1qOGmeoiU2BP2uyTlv31KJPDm08GwmPTTX8fi3LV
                                ) ; key id = 44494
groupX.odslab.se.       120 IN DNSKEY 257 3 8 (
                                AwEAAc6Wk/UqaEMaytXWL2y25I0Z8Uuubnkrufa JEEBw
                                niObHaNGMscp5I5207ScB6L70DJS46S9bA4k8mbcRNPA
                                Vi0OQVz1kFTTNt45XzYQ7yaQJyobQdFtVq8TXtaFPiFP
                                S7nz7ga8/HVW8VNRp4H5iajsgh4LCX+399tJX+rk613R
                                tbnHVvZPOUiuZNFqZLOkbzGtNRbl4UvoRQi5q+tjV/ow
                                cUkn8tljQGPpTe/HLImUT+MrftnY6m8jvgO+qhd2o/1Z
                                6XZcVBuDB+UGrhFcU72HmeKfQHMtCuGZhmWOcOymPcDJ
                                12ONkBqgj28Cu/4Kr44DMTu4q2ax07dDOfSyKqM=
                                ) ; key id = 62246
groupX.odslab.se.       120 IN RRSIG DNSKEY 8 3 120 20110628103715 (
                                20110628083552 62246 groupx.odslab.se.
                                Tw32FOW95e86g0FYxyXu3nDQNTdAELxVhg4BVoRA2RWx
                                iAgkZk/XQRUfozjd/qNNjrIA2+a9wwrvLWokRB6xzSTR
                                bwx199Mu8Xj9p9Q8CbzCvbvHPtRqPgf6Mto9jjlUaSK4
                                NlNQWg/qfsLvkvxRpdE4g9Xac3b71TPuylQSovvARR0v
                                4rJ4zmBdomdQHjtwOuQ4GeVfpgKqFCqa8HFK8D20Kmjk
                                56a7rbe6UWt5hHMjQfys3NfvulFAdCTW0Rbikss7YQMw
                                j6msmsRS8Zj+IlBbmku6RwxVxNF/ca09fuz4NhyOOSRP
                                2mBTBIwk+XcybA6vK5ofnrBTCSSoJOt4+g== )
groupX.odslab.se.       60 IN NSEC3PARAM 1 0 5 3A5BF749D1330DE3
groupX.odslab.se.       60 IN RRSIG NSEC3PARAM 8 3 60 20110628103502 (
                                20110628083552 44494 groupx.odslab.se.
                                Gvy1AOrm6dENvVUke1Ck3KmjB5W1mbvIsFdvm2p2MfZa
                                msgUJNJ0sT6R3jIyRIvc+6T3jADDHGpvr6ILLnWySFRb
                                9efAn/SDt060N3YsU6emv5iAh/TRbo7g8UNtokm1TAds
                                5rZ187cOo3yqQ05qBSTVo8wCcF1HS6+htEt+vQs= )
www.groupX.odslab.se.   60 IN CNAME nsx.odslab.se.
www.groupX.odslab.se.   60 IN RRSIG CNAME 8 4 60 20110628103414 (
                                20110628083552 44494 groupx.odslab.se.
                                BAs7KPVdwoPeC9isn/N00dV2OB62sSjbQS65r6h8EOGF
                                ToRqd6wRpd8OhNSNrJNn7ycH61m2j71WhE00fsMLA1T6
                                vxGKVcK6IeH+7Vpu4bgnH93jq8f3TftaiR22bYNl+Y9Q
                                Y7PHNFcmZ0PmoqVmilmtJdpn+YNjUJ5a+Riwojo= )
7oreb1sb9elhfqfp53bqqde6bcdm5eo3.groupx.odslab.se. 60 IN NSEC3 1 0 5 3A5BF749D1330DE3
OTANAROMKJB00QC2G6K2IT2GU2SB4DOA CNAME RRSIG
7oreb1sb9elhfqfp53bqqde6bcdm5eo3.groupx.odslab.se. 60 IN RRSIG NSEC3 8 4 60
20110628103552 (
                                20110628083552 44494 groupx.odslab.se.
                                azU2yBsLQNXANwyTxosI4hwf6JPfV5XKNdPtQzGprShE
                                w6N/sDG9QzMJjlQrPW82rY2SYl7xGJMBGdfsGVBZJJQ4
                                nXBmwnjT5Grm9k/a0hyCmYYAHzoq4ixV5fLDYrH8af/u
                                uvoFs90vJlN4OMbHNJUrNSsCsJRzps/k0/aH+0w= )
otanaromkjb00qc2g6k2it2gu2sb4doa.groupx.odslab.se. 60 IN NSEC3 1 0 5 3A5BF749D1330DE3
7OREB1SB9ELHFQFP53BQQDE6BCDM5EO3 NS SOA RRSIG DNSKEY NSEC3PARAM
otanaromkjb00qc2g6k2it2gu2sb4doa.groupx.odslab.se. 60 IN RRSIG NSEC3 8 4 60
20110628103526 (
                                20110628083552 44494 groupx.odslab.se.
                                QLlN/6CjlkU609P9/AntqRFHWAKJ8PUIS53HOZfN9D6P
                                PZEr/7dd+jlv2sgXmIYx/0VXySr4Bafgm8+k0fwEU+JY
                                TjmfkLUOD6O9DOQ/RqNtLp5HFH6TLMZxO7VdFr9vEZq1
                                5UIUQjIFT2+aQR3Dd/QMq26ysHGqOApSH/wkq6Y= )
```

Fingerprints

- A fingerprint is a checksum of a key. Fingerprints are often published instead of a key because it is much shorter than a key, and more easy to read.

```
BEAAAAPFUp17Etwawvfg7DV5k7mkdLGn42PcFcXyXOWr
rStBNWF2q6af2WOxMwlPqPb8bBKmm5QZErTZLuhgDVE8
KuPdnsxF90+pV2y9eB3+FIjDjQfo1xKcxAjRMaKkSrCA
WRA0PplQu2AfZW7q/MZK3O6uCwqp7xv4/nblU2PoVKpn
KXX6xkIhfbM/K/jnBJqprmBfzR+WcFLuP56Bf49/Vdv7
LRnDjuXWoRQ7gu7/W72fzXwOwy5DqRf0G7iKIltEZOjp
M8nROvp3w35naNLC6o0bbgw1MlE3sOAn8IiLLw+Kn7kJ
  kfB1uGPUzqdf1wSx0wcfBaRnnPQdlnH80OGRBdDN
```

```
A1B8B850CAA2D3C595D5617DB5ADE18989CC542CD15B9B0236E7D3752AAC2946
```

DS records

- DS - Delegation Signer
- A DS record (the hash of the DNSKEY) is published at the parent zone to delegate trust to the child zone.
- This is what is published for opendnssec.se at .se:

```
opendnssec.se.   IN  DS  27295 5 1
5AEF372D65BC594A7AF5E0E77CDDA55E0C43A56A
opendnssec.se.   IN  DS  27295 5 2
A1B8B850CAA2D3C595D5617DB5ADE18989CC542CD15B9B0236E7D3752A
AC2946
```

- Two DS records - two algorithms are used for .se, SHA-1 and SHA-256
- The DS are signed by the parent

Key rollovers

- Key can be removed and added
- The rollover process must follow a set of rules
- Different states
  - E.g. pre-published, active, and post-published
- The software will assist you

Components in DNS

- Name server
  - The signer can be integrated in the name server or act as a separate component in the distribution chain.
- Resolver
  - Needs to understand DNSSEC and be configured with a trust anchor.

Resolver

- BIND
- Unbound
- Windows 2008 R2
- …

Name server

- BIND
- NSD
- Windows
- PowerDNS
- djbdns
- …

DNSSEC signing software

- BIND
- OpenDNSSEC
- PowerDNS
- …

DNSSEC appliances

- Secure64
- Infoblox
- Xelerance Corp
- Men & Mice
- BlueCat Networks

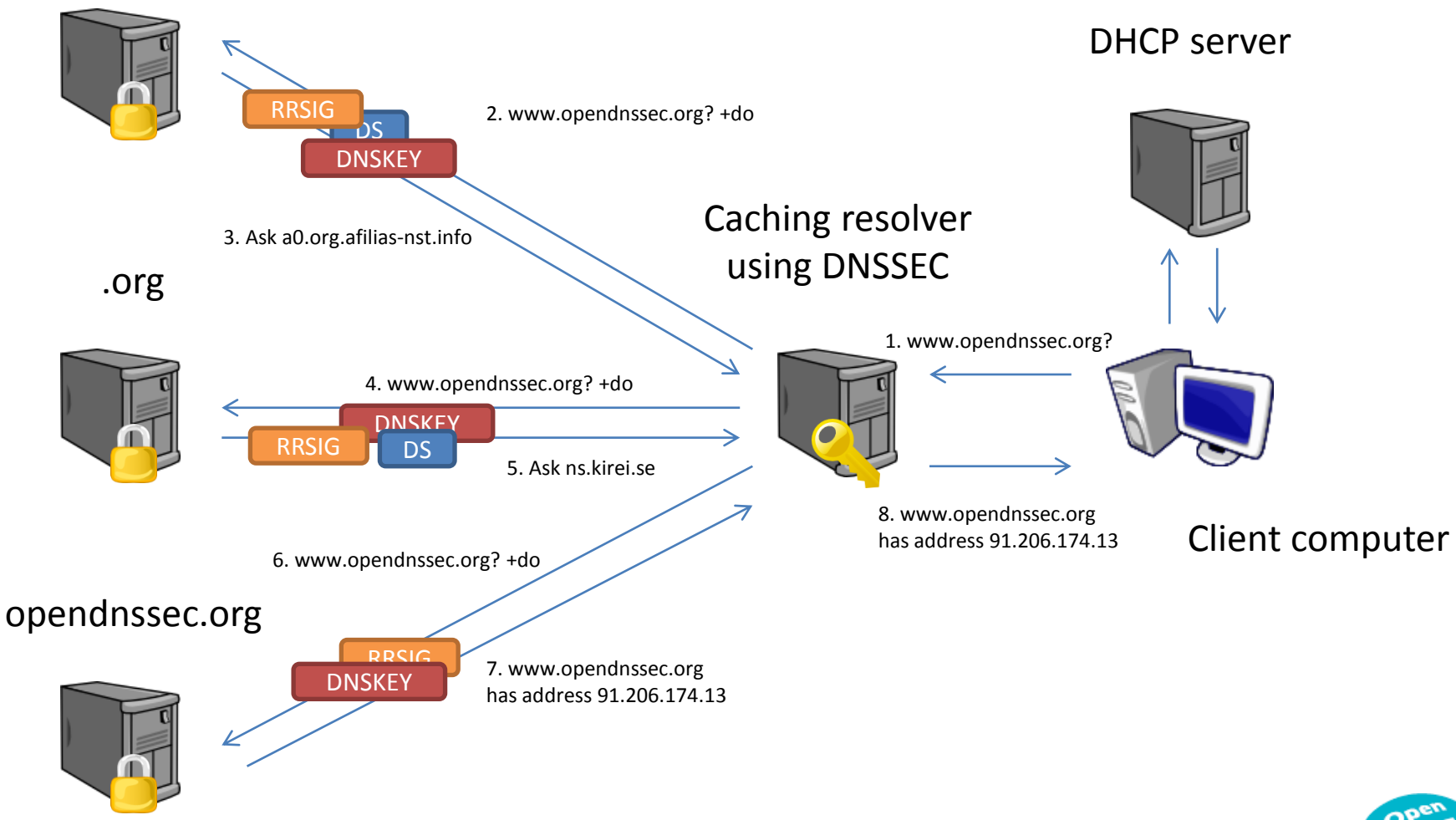
http://www.iis.se/docs/DNSSEC-Admin-tools-review-Final.pdf

Start verifying signatures

- Get the root trust anchor from IANA.
- Verify its authenticity
- Configure BIND:

```
managed-keys {
    <INSERT KEY>
};
options {
    dnssec-enable yes;
    dnssec-validation yes;
};
```

Resolving DNS



OpenDNSSEC training

OpenDNSSEC Architecture

What?

- OpenDNSSEC is a zone signer that automates the process of keeping track of DNSSEC keys and the signing of zones.

Why?

- The available DNSSEC tools were lacking:
  - Good key management
  - Policy handling
  - Hardware acceleration
  - Etc.
- DNSSEC should be easy to deploy
- Increase the number of DNSSEC users
- Experience from previous DNSSEC operations

Who?

kirei

NLnet Labs

nominet

SIDN

SURF NET

.se

sinodun

.ca | Canadians Connected Canadiens branchés

(The logos belongs to the individual organizations and are not covered by this CC license)

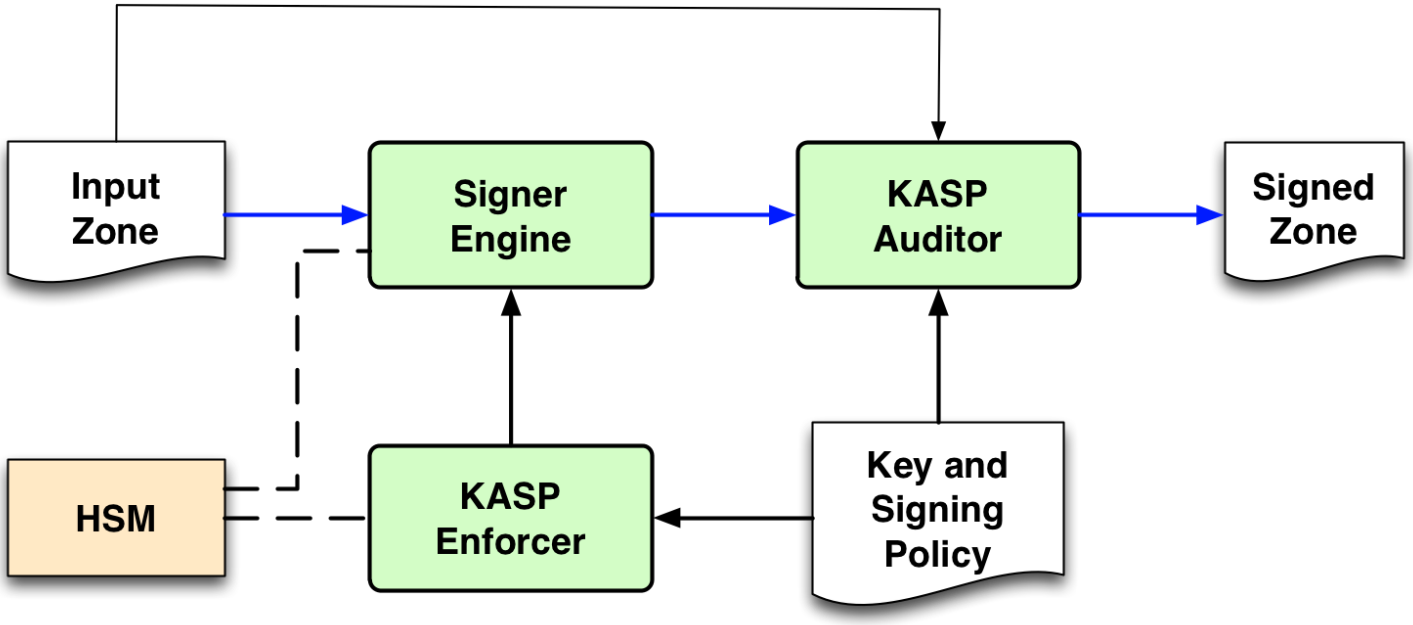
About OpenDNSSEC

- Simplifies the process of signing one or more zones
- Reducing the work load on the system administrator
- Open source software with a BSD license
- Simple to integrate into existing infrastructure
- Key storage and hardware acceleration using PKCS#11

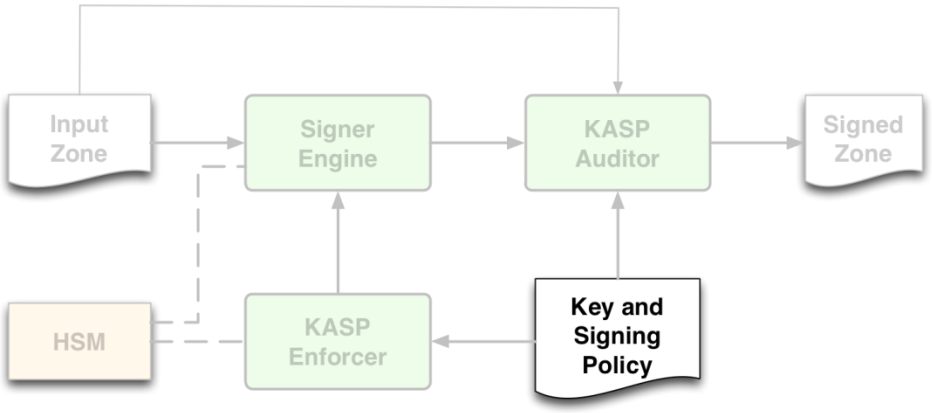
Bump-in-the-Wire

- In many cases, anticipate that OpenDNSSEC will be employed on a system between a hidden and public master.

Architecture

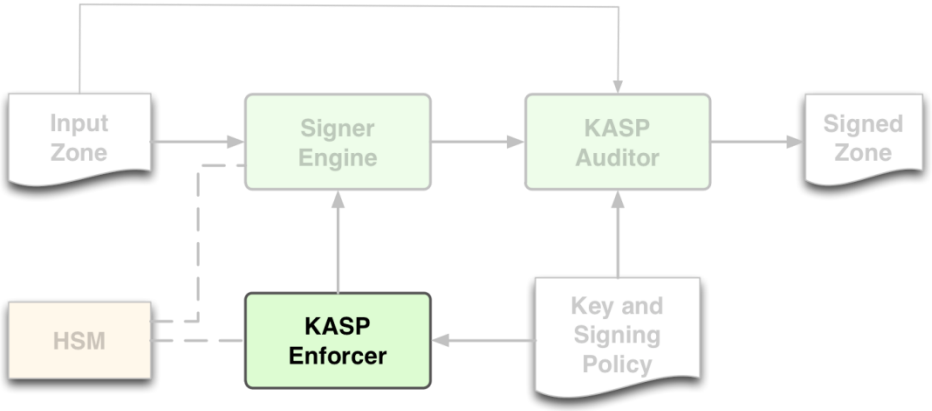


Key and Signing Policy



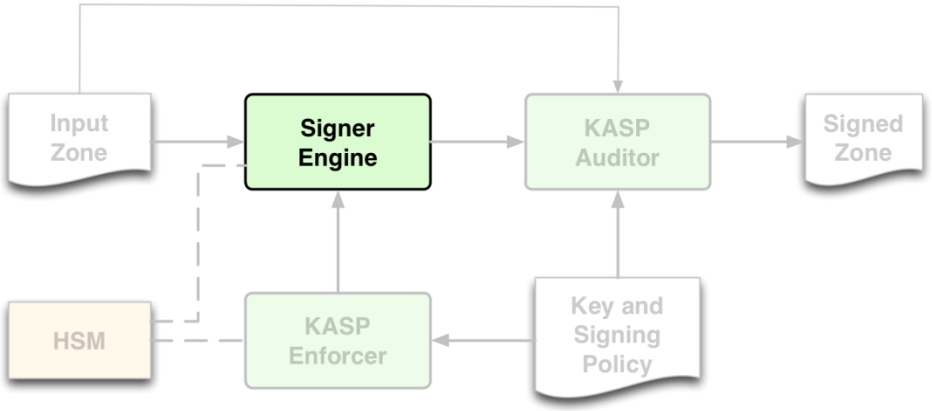
- How to sign a zone is described by a policy
- Allows choice of key strengths, algorithm, key and signature lifetimes, NSEC/NSEC3, etc.
- Can have anything between one policy for all zones to one policy per zone.

KASP Enforcer



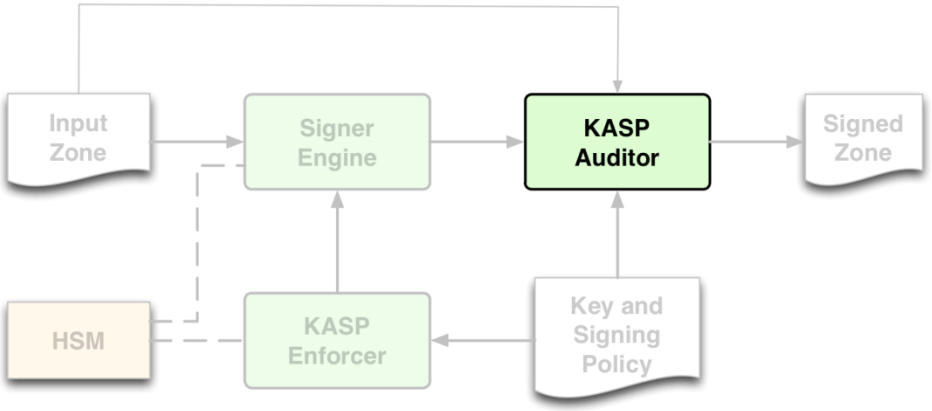
- Handles the management of keys:
  - Key creation using HSM
  - Key rolling
- Chooses the keys used to sign the zone.

Signer Engine



- Automatic signing of the zones
  - Can reuse signatures that are not too old
  - Can spread signature expiration time over time (jitter)
- Maintains the NSEC/NSEC3 chain
- Updates SOA serial number

KASP Auditor



- Checks that the Signer and Enforcer work the way they are supposed to, e.g.
  - Non DNSSEC RRs are not added or removed
  - Policy is being followed
- Can stop the zone distribution if needed
- Written in Ruby

Daemons

- Enforcer
  - ods-enforcerd
- Signer Engine
  - ods-signerd

CLI

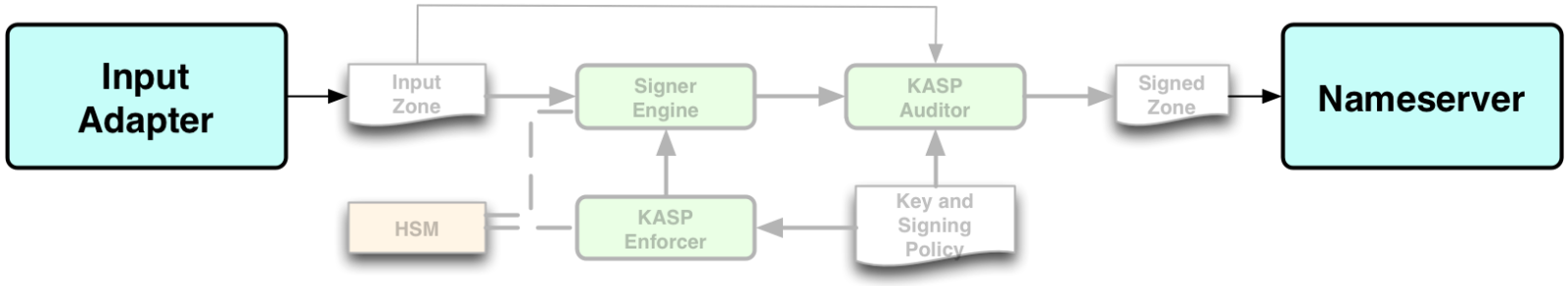
- General
  - ods-control
  - ods-kasp2html
- Enforcer
  - ods-ksmutil
- Signer Engine
  - ods-signer
- Auditor
  - ods-auditor
  - ods-kaspcheck
- HSM
  - ods-hsmspeed
  - ods-hsmutil

HSMs

- Why should you use one?
  - Security (FIPS)
    - The private keys are stored securely in the HSM
    - You know where your keys are
  - Speed
    - 1 – 13,000 signatures per second
- Are they expensive?
  - $50 - $50,000
- Remember to protect the host
  - Garbage in -> Garbage out

- SoftHSM is a software-only implementation of an HSM using the PKCS#11 interface
- Can be used to test the PKCS#11 interface without buying a real HSM.
- Uses Botan and SQLite.
- SoftHSM makes it possible to use OpenDNSSEC in a software-only environment.

Input and Output Adapters



- Input adapter supplied as part of OpenDNSSEC - accepts AXFRs, responds to NOTIFYs.
- Output adapter not supplied - any preferred nameserver can be used (BIND, NSD, etc.)
- Can configure command to be used to reload zone.

OpenDNSSEC training

Installing OpenDNSSEC

Hardware

- CPU
  - Worker threads – Handle multiple zones at a time
  - Signer threads – Maximum performance from the HSM
- HDD
  - Backup copy of the unsigned and the signed zones
- Memory
  - The signed zones are stored in memory
  - May be doubled temporarily before the changes are committed

Platform support

- OpenDNSSEC has been tested on various platforms:
  - Debian
  - FreeBSD
  - Gentoo
  - Mac OS X
  - NetBSD
  - OpenBSD
  - Red Hat Enterprise Linux
  - Solaris
  - Ubuntu

Pre-built binaries

- OpenDNSSEC are or will be available as packages for the following systems:
  - Debian
  - FreeBSD
  - Gentoo
  - NetBSD
  - Ubuntu

Dependencies

- ldns
- libxml2, libxml2-dev, libxml2-utils
- ruby, rubygems, dnsruby, libopenssl-ruby
- sqlite3, libsqlite3, libsqlite3-dev
- (mysql-client, libmysqlclient15, libmysqlclient15-dev)
- libbotan (SoftHSM)

Obtaining the source code

- Tarballs:
  - www.opendnssec.org
- SVN:
  - `svn co http://svn.opendnssec.org/ ods-svn`

Building the code

- Follow the lab instructions on how to build the code

OpenDNSSEC training

Hardware Security Modules

WHAT IS AN HSM?

What is an HSM?

- Protected keystore
  - Private keys can never be extracted in clear
- Crypto hardware
  - Sometimes increases speed (but not always)
- Well-defined software interface

Protected keystore

- Keys stored in tamperproof memory
  - If you mess with the chip, the device will (try to) detect it and zeroize
- Implemented using
  - Covering components in epoxy
  - Thin wires covering sensitive components

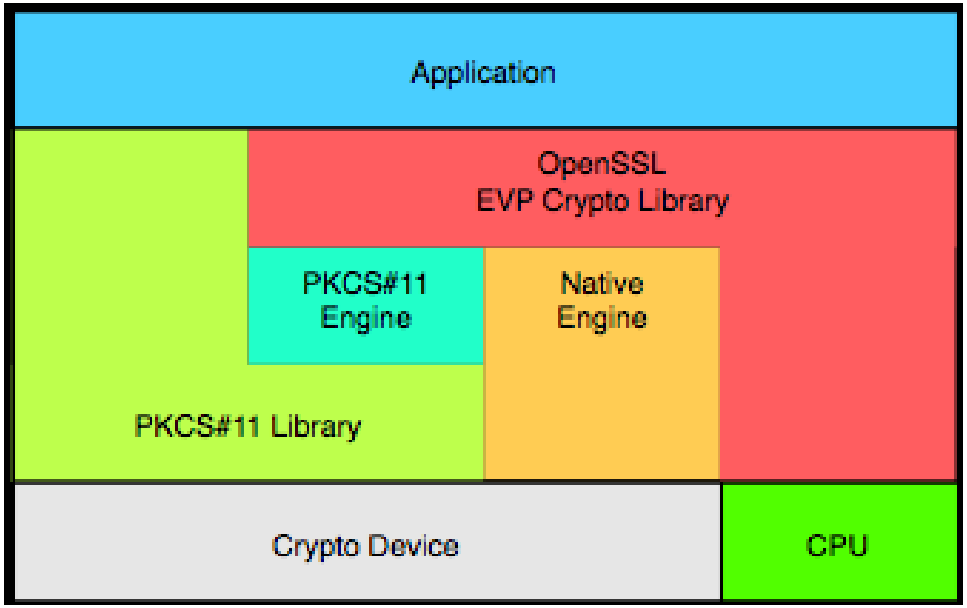
Crypto hardware

- Hardware to assist accelerate symmetric and asymmetric crypto
  - RSA, DSA, AES, 3DES
  - Good random number generator
- Hashing is often implemented in the host

API

- PKCS#11 (aka Cryptoki)
- OpenSSL Engine
- Microsoft CAPI
- Java Cryptography Extension

Stacked APIs are possible…



PKCS#11

- E.g.:
  - C_Initialize
  - C_GetSlotList
  - C_OpenSession
  - C_Login
  - C_GenerateKeyPair
  - C_FindObjectsInit, C_FindObjects, C_FindObjectsFinal
  - C_SignInit, C_Sign
  - C_Finalize

WHY USE AN HSM?

What is the risk?

- Keys can be compromised by…
  - Compromised hosts
  - Disgruntled staff
  - Math

How to lower the risk?

- Protect the host itself
  - But some sort of remote management is usually needed anyway
- Protect the private keys
  - Move keys to HSM

Residual risk

- Keys can still be misused
  - If you can use a key, you can also misuse it
- Garbage in -> Garage out
  - If you feed it a bad zone – the result is still a signed bad zone

Increase trust?

- Using an HSM increases trust – Why?
  - Standards compliance
  - Verifiable security – e.g. FIPS 140-2
- Also provides a clean cut between keystore and signing software
  - You know where your keys are (and not are)

THE BUYER'S GUIDE TO HARDWARE SECURITY MODULES

Types of HSMs

- Local interface – e.g. PCI cards
- Remote interface – e.g. Ethernet
  - Sharable between multiple hosts
- Smart cards
- USB tokens
  - usually a smart card with integrated reader

Algorithms and key sizes

- What algorithms are supported
  - RSA recommended, DSA and GOST optional
- What key sizes are supported
  - Minimum key size ≤ 1024 bits recommended
  - Maximum key size ≥ 2048 bits recommended

Capacity

- How many keys can be stored?
- Where are the keys stored?
  - Internal keystore
  - External keystore (encrypted by a master key)

API

- What API do you need?
  - PKCS#11, OpenSSL, MS-CAPI, JCE
- What platforms are supported?
  - Mind details like Linux kernel versions, distributions etc.

Speed

- Signing speed – RSA
  - Usually measured in 1024-bit signing operations (with public exponent 3 or 65537) per second.
- Key generation speed – RSA
  - Usually the average key generation time for 1024-bit and 2048-bit keys per second.

Security certifications

- FIPS 140-2
  - Federal Information Processing Standard
- CC-EAL
  - Common Criteria Evaluation Assurance Levels

FIPS 140-2

Level	Requirement
1	Basic security requirements
2	Tamper evidence, user authentication
3	Tamper detection/resistance, data zeroisation, splitting user roles
4	Very high tamper detection/resistance, Environmental protection

CC-EAL

- What Protection Profile (PP) has been used for the Target of Evaluation (TOE)?
  - CMCKG-PP – Key Generation
  - CMCSO-PP – Signing Operations

Key backup

- How do you backup your keystore?
- Can you restore a backup elsewhere?
  - e.g. on a hot-standby site
- Split key backup possible?
- Well-known backup format?

OPENDNSSEC AND HSMS

HSMs

- The following Hardware Security Modules (HSM) has been confirmed to work with OpenDNSSEC:
  - AEP Keyper
  - Aladdin eToken
  - Athena Smartcard Solutions IDProtect
  - OpenSC Smart Cards
  - Safenet Luna SA
  - Sun Crypto Accelerator 6000 (SCA/6000)
  - Thales nShield Connect
  - Utimaco SafeGuard CryptoServer

Review

- Conducted a review of four different HSM:s
  - AEP Keyper v2
  - SafeNet Luna SA 4.4
  - Thales nShield Connect 6000
  - Utimaco CryptoServer Se1000

http://www.opendnssec.org/wp-content/uploads/2011/01/A-Review-of-Hardware-Security-Modules-Fall-2010.pdf

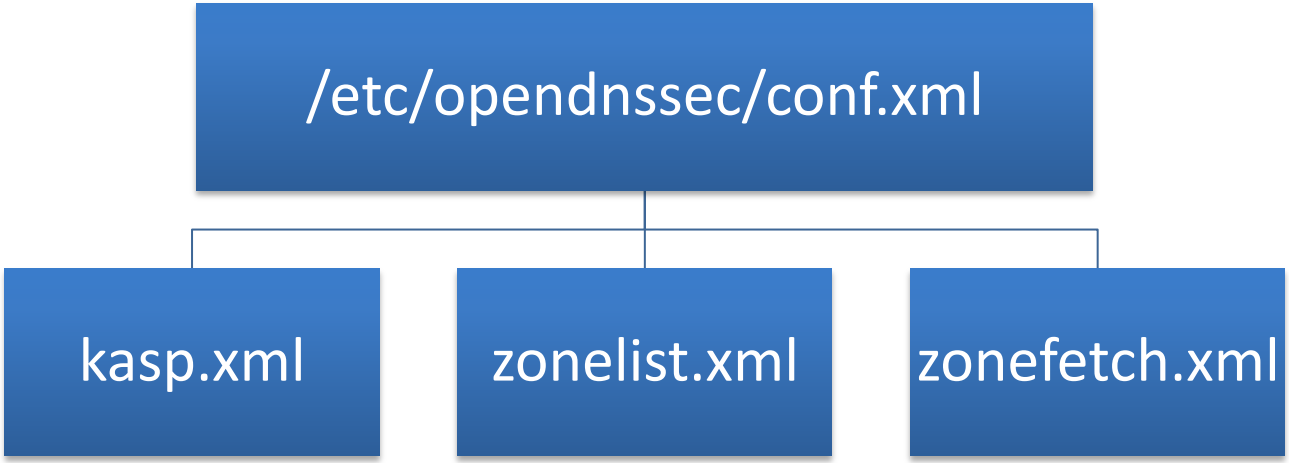
OpenDNSSEC training

OpenDNSSEC configuration

XML-files

- conf.xml
  Used for overall configuration of the system
- kasp.xml
  Defines the various policies for signing zones
- zonelist.xml
  Zones that will be signed using a policy
- zonefetch.xml
  for transferring/fetching zones

XML-files



P[n]Y[n]M[n]DT[n]H[n]M[n]S

- OpenDNSSEC is about durations (periods), not about absolute times.
- The format of periods is as above
  - P1DT12H is 1 day and 12 hours
- No clue about Gregorian Calendar
  - P1M is considered 1 month (always 31 days)
  - P1Y is considered 1 year (always 365 days)

conf.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id: conf.xml.in 5227 2011-06-12 08:51:24Z jakob $ -->
```

- Preamble... It's what you get when you use XML

conf.xml

- Configuration contains
  - RepositoryList
  - Common
  - Enforcer
  - Signer
  - Auditor

```
<Configuration>
  <RepositoryList>
    ....
  <RepositoryList>
  <Common>
    ....
  </Common>
  <Enforcer>
    ....
  </Enforcer>
  <Signer>
    ....
  </Signer>
  <Auditor>
    ....
  </Auditor>
</Configuration>
```

conf.xml

```
<RepositoryList>
  <Repository name="SoftHSM">
    <Module>/usr/local/lib/libsofthsm.so</Module>
    <TokenLabel>OpenDNSSEC</TokenLabel>
    <PIN>1234</PIN>
    <!-- <Capacity>1000</Capacity> -->
    <!-- <RequireBackup/> -->
    <SkipPublicKey/>
  <Repository>
  ...
</RepositoryList>
```

name, also used in kasp.xml

- Defines where private keys live
- You need at least one but you can have more

conf.xml

```
<Common>
  <Logging>
    <Verbosity>3</Verbosity>
    <Syslog><Facility>local0</Facility></Syslog>
  </Logging>

  <PolicyFile>/etc/opendnssec/kasp.xml</PolicyFile>
  <ZoneListFile>/etc/opendnssec/zonelist.xml</ZoneListFile>

  <!—
    <ZoneFetchFile>/etc/opendnssec/zonefetch.xml</ZoneFetchFile>
  -->
</Common>
```

- This elements provides pointers to other configuration files and some settings shared by all components such as logging

conf.xml

```
<Enforcer>
  <!—
    <Privileges>
      <User>opendnssec</User>
      <Group>opendnssec</Group>
    </Privileges>
  -->

  <Datastore><SQLite>/var/opendnssec/kasp.db</SQLite></Datastore>
  <Interval>PT3600S</Interval></Enforcer>
  <!-- <ManualKeyGeneration/> -->
  <!-- <RolloverNotification>P14D</RolloverNotification> -->
  <!-- <DelegationSignerSubmitCommand>/usr/local/sbin/eppclient
       </DelegationSignerSubmitCommand> -->
</Enforcer>
```

- Can also use MySQL

conf.xml

```
<Signer>
  <!--  <Privileges><User>opendnssec</User><Group>opendnssec</Group>
        </Privileges> -->

  <WorkingDirectory>/var/opendnssec/tmp</WorkingDirectory>
  <WorkerThreads>8</WorkerThreads>
  <SignerThreads>8</SignerThreads>

  <!-- <NotifyCommand>rndc reload %zone</NotifyCommand> -->
</Signer>
```

- The Signer will need a place to put temporary files and may start multiple threads.
- After the Signer is done you may want to kick your name server for a reload

conf.xml

```
<Auditor>
  <!—
    <Privileges>
      <User>opendnssec</User>
      <Group>opendnssec</Group>
    </Privileges>
  -->

  <WorkingDirectory>/var/opendnssec/tmp</WorkingDirectory>
</Auditor>
```

- The Auditor will also need a place to put temporary files

kasp.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id: kasp.xml.in 5227 2011-06-12 08:51:24Z jakob $ -->
```

- Key and Signature Policy is documented in here

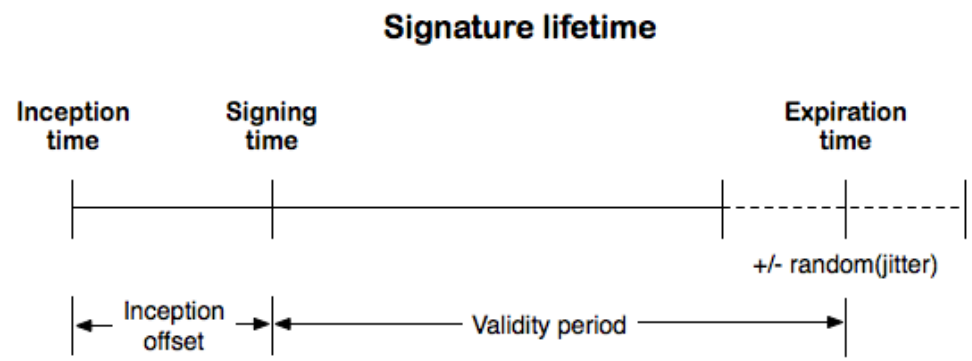
kasp.xml

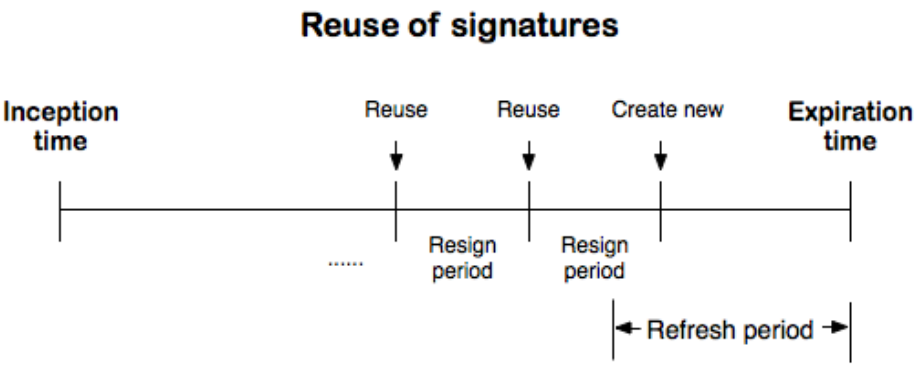
- KASP contain one or more policies
- Policy contains
  - Description
  - Signatures
  - Denial
  - Keys
  - Zone
  - Parent
  - Audit

```
<KASP>
  <Policy>
    <Description>
      ....
    </Description>
    <Signatures>
      ...
    </Signatures>
    <Denial>
      ...
    </Denial>
    <Keys>
      ...
    </Keys>
    <Zone>
      ...
    </Zone>
    <Parent>
      ...
    </Parent>
    <!-- <Audit/> -->
  </Policy>
  <Policy>
    ....
  </Policy>
</KASP>
```

kasp.xml

```
<Signatures>
  <Resign>PT2H</Resign>
  <Refresh>P3D</Refresh>
  <Validity>
    <Default>P7D</Default>
    <Denial>P7D</Denial>
  </Validity>
  <Jitter>PT12H</Jitter>
  <InceptionOffset>PT3600S</InceptionOffset>
</Signatures>
```





kasp.xml

```
<Denial>
  <NSEC3>
    <!-- <OptOut/> -->
    <Resalt>P100D</Resalt>
    <Hash>
      <Algorithm>1</Algorithm>
      <Iterations>5</Iterations>
      <Salt length="8"/>
    </Hash>
  </NSEC3>
</Denial>
```

- Denials defines parameters for Denial of Existence
- Use <NSEC/> for NSEC

kasp.xml

```
<KEYS>
  <TTL>PT3600S</TTL>
  <RetireSafety>PT3600S</RetireSafety>
  <PublishSafety>PT3600S</PublishSafety>
  <!-- <ShareKeys/> -->
  <Purge>P14D</Purge>
  ......
```

- The KEYS element defines the lifetimes of keys
- The TTL ends up in the DNSKEY RRset
- Retire and Publish Safety are safety margins for during key rollover
- Purge is when to remove keys

kasp.xml

```
<KEYS>
  ......
  <KSK>
    <Algorithm length="2048">7</Algorithm>
    <Lifetime>P1Y</Lifetime>
    <Repository>SoftHSM</Repository>
  </KSK>
  ......
</KEYS>
```

Repository from conf.xml

- KSK sets KSK parameters for the current policy

kasp.xml

```
<KEYS>
  ......
  <ZSK>
    <Algorithm length="1024">7</Algorithm>
    <Lifetime>P30D</Lifetime>
    <Repository>SoftHSM</Repository>
    <!-- <ManualRollover/> -->
  </ZSK>
</KEYS>
```

Repository from conf.xml

- ZSK sets ZSK parameters for the current policy

kasp.xml

```
<Zone>
  <PropagationDelay>PT43200S</PropagationDelay>
  <SOA>
    <TTL>PT3600S</TTL>
    <Minimum>PT3600S</Minimum>
    <Serial>unixtime</Serial>
  </SOA>
</Zone>
```

- The propagation delay is the time it takes for a zone to get to the complete set of name servers. Should be larger than the SOA refresh and not be larger than the SOA expiry parameter
- keep, unixtime, datecounter, counter

kasp.xml

```
<Parent>
  <PropagationDelay>PT9999S</PropagationDelay>
  <DS>
    <TTL>PT3600S</TTL>
  </DS>
  <SOA>
    <TTL>PT3600S</TTL>
    <Minimum>PT3600S</Minimum>
  </SOA>
</Parent>
```

- Parent timing is important for maintaining the Chain of Trust.
- Look at the parental parameters and configure them in here
- Note that your parent may change its settings so now and then

kasp.xml

```
<Audit>
  <!-- <Partial /> -->
</Audit>
```

- If this element is present than all zones according to the current policy will be ‘audited’ after they are signed
  - May take a long time
  - May run out of memory
- Independent code path
- Not always that liberal in parsing ‘exotic’ RRs

Configuration

- We configured conf.xml and kasp.xml
- Remember that you can have multiple policies
  - One HSM slot serving 100 static zones with 1 private key
  - A SoftHSM for zone signing and a HSM for key signing
  - Zones with or without parents
  - Zones with different parents (.se and .org)
- We have to tie the policies defined in kasp.xml to the zones we want to sign

zonelist.xml

```
<ZoneList>
  <Zone name="example.com">
    <Policy>default</Policy>
    <SignerConfiguration>/var/opendnssec/signconf/example.com.xml
    </SignerConfiguration>
    <Adapters>
      <Input>
        <File>/var/opendnssec/unsigned/example.com</File>
      </Input>
      <Output>
        <File>/var/opendnssec/signed/example.com</File>
      </Output>
    </Adapters>
  </Zone>
  ...
</ZoneList>
```

zonefetch.xml

- The configuration to use if the zones will be fetched by using AXFR
- This is documented online.

OpenDNSSEC training

Key states

Key states

- Extra precaution needs to be taken because of the DNS caches
- TTL and other timing attributes creates a delay before all information has propagated
- Use key states to get control of this process

Key states

- Publish
- Ready
- Active
- Retire
- Dead

- DSSub
- DSPublish

OpenDNSSEC training

Key rollovers

DNSSEC Key Timing Considerations

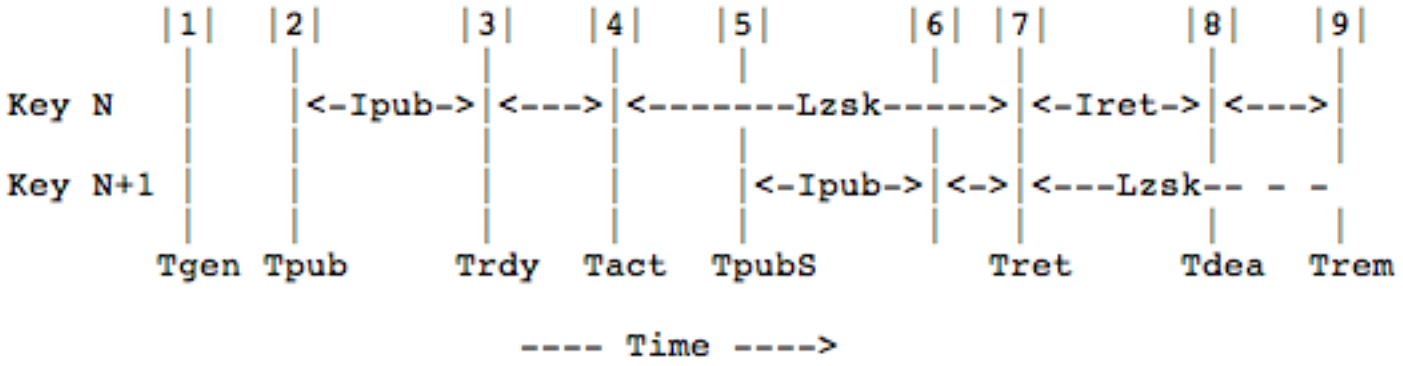
- A draft describing the process of rolling keys.

- http://tools.ietf.org/html/draft-ietf-dnsop-dnssec-key-timing-02

Rollover mechanisms

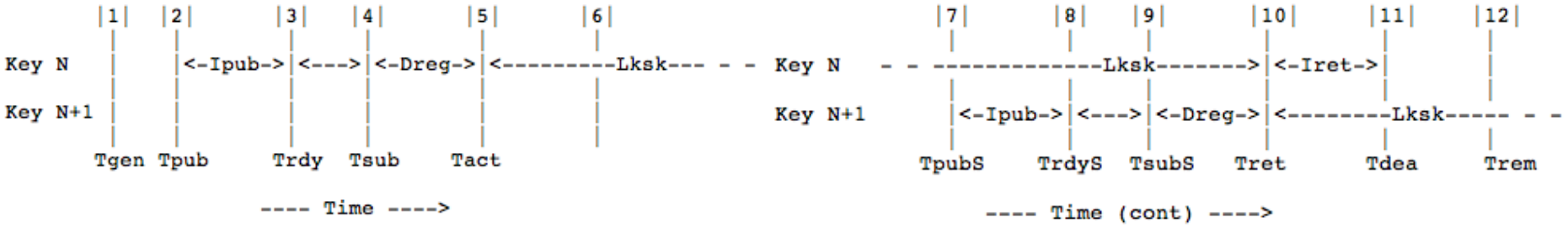
ZSK Method	KSK Method	Description
Pre-Publication	N/A	Publish DNSKEY before the RRSIG
Double-Signature	Double-Signature	Publish DNSKEY and RRSIG at the same time. For a KSK, this happens before the DS is published
Double-RRSIG	N/A	Publish RRSIG before the DNSKEY
N/A	Double-DS	Publish DS before DNSKEY
N/A	Double-RRset	Publish DNSKEY and DS in parallel.

Pre-Publication ZSK rollover



- First key: Ipub = Dprp + min(TTLsoa, SOAmin)
- Future keys: Ipub = Dprp + TTLkey
- TpubS <= Tact + Lzsk - Ipub
- Iret = Dsgn + Dprp + TTLsig

Double-Signature KSK rollover



- Ipub = Dprp + TTLkey
- TpubS <= Tact + Lksk - Dreg - Ipub
- Iret = DprpP + TTLds

Default KASP

- The default KASP will work in many cases
- But verify that the values works in your environment

OpenDNSSEC training

Testing

Testing

- Always verify that the zone works before publishing your first DS.
- There are various tools that can help.
- Can also trouble shoot any problems you might have.

DNSCheck

- DNSCheck is a program that was designed to help people check, measure and hopefully also understand the workings of the Domain Name System, DNS.
- Open source software
- Available online and as a CLI
- Demo: http://dnscheck.iis.se/?setLanguage=en

DNSViz

- DNSViz is a tool for visualizing the status of a DNS zone.
- Demo: http://dnsviz.net/

OARC's DNS Reply Size Test Server

- DNSSEC required resolvers and the network to handle large packets
- This tool can show you what limitations there are

- Demo: `dig +short rs.dns-oarc.net TXT`

- https://www.dns-oarc.net/oarc/services/replysizetest

OARC's source port test

- Some resolvers do not randomize the source port of the DNS query
- Demo: `dig +short porttest.dns-oarc.net TXT`
- https://www.dns-oarc.net/oarc/services/porttest

DNSSEC-debugger

- An online tool to verify the trust chain

- Demo: http://dnssec-debugger.verisignlabs.com/

OpenDNSSEC training

Integration

Integration into an existing system

- Adding/removing zones
- Zone distribution
- Send the public keys to the parent zone

Adding/removing zones

- Edit the zone list
  - Update the information in zonelist.xml
  - Trigger OpenDNSSEC to re-read the zonelist (ods-ksmutil update zonelist)
- Or only use CLI
  - ods-ksmutil zone add --zone <name of zone>
  - ods-ksmutil zone delete --zone <name of zone>
  - If the extra arguments are not used, then the system defaults will be used
  - Will edit the zonelist.xml for you

Zone distribution

- OpenDNSSEC currently only support AXFR in, file in, and file out
- Remember to trigger OpenDNSSEC to re-read the zone file if you use file in
- Future versions will have better support
- You can use your favorite nameserver to serve the signed zone file
- Use <NotifyCommand>rndc reload %zone</NotifyCommand> in conf.xml

Sending keys to the parent zone

- Manually
  - Extract the keys from OpenDNSSEC or the signed zone
- Automatic
  - Use <DelegationSignerSubmitCommand> in conf.xml
  - OpenDNSSEC sends the current set of DNSKEY RR which should have a corresponding DS RR in the parent zone
  - A command which can receive DNSKEY RRset on STDIN
  - The command has to do its own conversion to DS RR
  - Write your own plugin or use the ones provided by OpenDNSSEC

Plugins

- EPP client
- simple-dnskey-mailer

OpenDNSSEC training

Monitoring

Why?

- We must have a zone with valid signatures and no missing data.
- Can be caused by various issues:
  - Configuration errors
  - Name servers not receiving updates
  - Unsynchronized clocks
  - Software bugs

What to monitor

- Signatures that are about to expire or is invalid
- Missing zone data
- Availability
- SOA Serial
- Policy compliance
- Etc.

Keep an eye on your system

- Active
  - Is part of your distribution chain
  - Can stop the distribution
- Passive
  - External monitoring
  - Can view the system from different points

Active monitoring

- The Auditor
- Internal scripts which check the zone before pushing the zone to the public name servers

Passive monitoring

- Monitor the system health
  - CPU load
  - Memory
  - Etc.
- Regularly perform queries against the public name server
- There are e.g. DNSSEC monitoring available for Nagios

OpenDNSSEC training

Disaster Recovery Plan

Disaster Recovery Plan

- DNSSEC requires more from your DNS operations.
- The time in DNSSEC is absolute and not relative.
- If something happens, you need to be able to act.
- You need to have a plan for different scenarios.

Backup

- Remember to create a backup of your environment.
  - KASP database
  - Keys
- The KASP database can be partially recreated, but requires a lot of work. Better to have a backup.
- Consult your HSM documentation on how to backup your keys.

Documentation

- Always have documentation on your environment.
  - System
  - Routines
  - Commands
- Easier for you to remember.
- Easier for others to work with the system.

Shared responsibility

- Share your knowledge with others in your organization.
- More should know how DNSSEC works.

Have a sane KASP

- It is good to have short lifetime on signatures from a security perspective.
- But can you fix the problem before the signatures expires?
- It is a trade-off between availability and integrity.

Going unsigned

- In the worst case scenario you might need to go unsigned.
  - Lost your keys, etc.
- Remove the DS from the parent zone.
- Must be done before the signatures expires.
- Remember to take TTL and propagation delay into account.

OpenDNSSEC training

Operational Practices

Algorithm

- Current recommendation is to use RSA/SHA-256
- SHA1 is becoming weaker
- SHA256 used by the root

Rolling KSK

- Different thoughts
  - Every 12 month
  - Roll when you “need” to
- Root will roll every 5th year

Rolling ZSK

- Current recommendation is every month
- Root is rolling every 3rd month

Single Type Signing Key

- One key acting as both KSK and ZSK
- Can be used when:
  - The exposure to risk is low (e.g. when keys are stored on HSMs).
  - One can be certain that a key is not used as a trust-anchor.
  - Maintenance of the various keys cannot be performed through tools.
  - The interaction through the registrar-registry provisioning chain, in particular the timely appearance of a new DS record in the 2011 parent zone in emergency situations, is predictable.
- Not yet supported by OpenDNSSEC

NSEC or NSEC3

- NSEC
  - When zone content is not highly structured or trivially guessable
  - Ease the work required by signers and validating resolvers
- NSEC3
  - Prevention of zone enumeration
  - Opt-out when the number of secure delegations is low

SOA Expire

- Always have valid signatures in your zone
- The zone should expire before the signatures
- SOA Expire < Signature Refresh Period

DNSSEC Policy & Practice Statement

- A framework for describing your DNSSEC Policy and operations
- Useful for relying parties when trusting your zone
- Also a good check list when deploying DNSSEC

- http://tools.ietf.org/html/draft-ietf-dnsop-dnssec-dps-framework-04
- https://www.iana.org/dnssec/icann-dps.txt

OpenDNSSEC training

Closing

Discussion

- Are you missing any functionality in the software?

Discussion

- Did we meet your expectations?
- If not, what more would you like to know?

Thank you

- The material is available online
  - www.opendnssec.org