Secure DNS

"Wer, wie, was − wieso, weshalb, warum?" *

Practical Linux

Giessen
27. Oktober 2007

Holger.Zuleger@hznet.de

* Titellied der Sesamstrasse

Agenda

- Einführung
  - Warum DNSsec ? − Historie
  - Abkürzungen und Terminologie
  - DNS Namensauflösung − Ein Überblick
- Signierte Zonen
  - Schlüsselerzeugung / Signieren der Zone
  - Key- und Zone Signing Keys
  - Signieren des Parent / Secure Delegation
- Werkzeuge
  - Administrative Aufgaben
  - Beispiel: Zone Key Tool
- Secure Resolver
  - The big view: Chain of Trust
  - Resolver Konfiguration (Trusted-Keys)
  - Secure Root Testbed

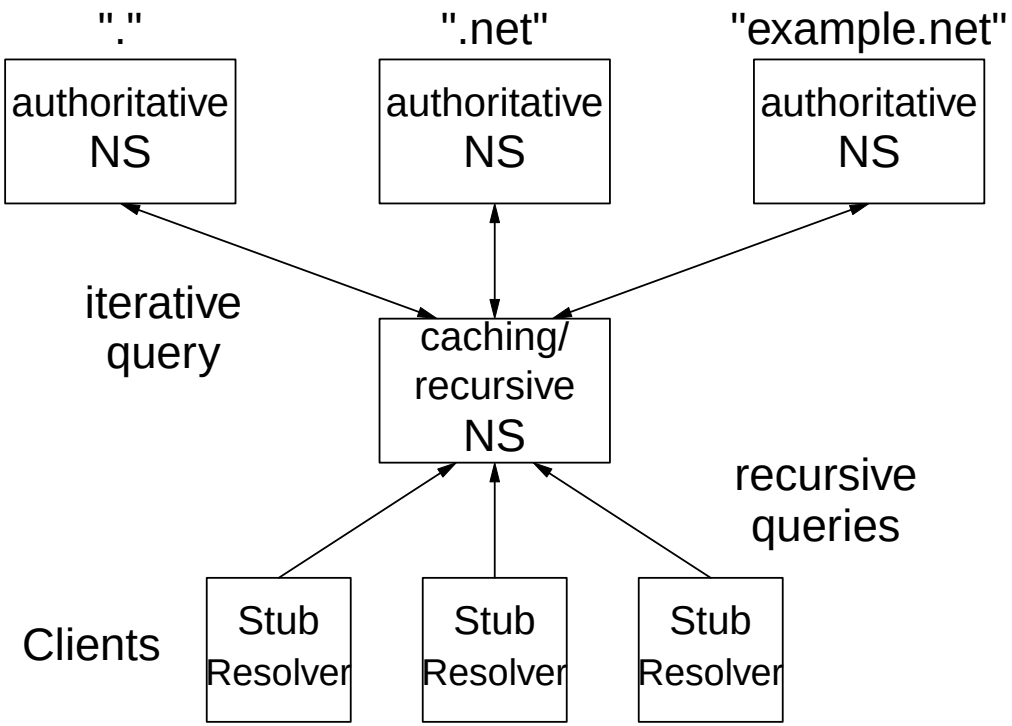
Warum DNSsec? − Historie

- DNS ist eines der älteren Internet Protokolle
  Entwickelt 1983 von P. Mockapetris, formale Definition RFC 1034/1035 (1986)
- Viele ältere Protokolle haben Sicherheitsmängel
  1990 beschreibt Steven Bellovin Angriffsszenarien gegen DNS
- Entwicklung von DNSsec startet 1995
- Formale Definition 2005 in RFC4033, RFC4034, RFC4035
  Implementiert ab BIND 9.3 und NSD 2.1.x
- Bedeutung des DNS nimmt zu
  - Immer mehr Dienste benötigen DNS
    (Anti SPAM (DKIM/SPF), ENUM, SIP, SSH-Fingerprints, IPSECKEY)
  - Immer mehr Dienste benötigen mehrstufige DNS Auflösung
    (MX, SRV, NAPTR)
  - Phishing/Pharming
- Bekannte Sicherheitslöcher sind nicht immer gefixed...
  „BIND 9: cryptographically weak query ids“ vom 24. Juli 2007

Abkürzungen und Terminologie

- Nameserver
  - Authoritative Nameserver (Hosted Domains)
  - Recursive oder Caching Nameserver (Resolver)
- Stub Resolver
  - Endsystem; sendet recursive Anfragen an Caching Resolver
- Domain / Zone
  `net`, `example.net`, `sub.example.net` (`www.sub.example.net`)
- Validator (Überprüft die DNSsec Signaturen)
  - Im Caching Resolver oder im Endsystem (Stub resolver) integriert
- Resource Record (RR)
  - Datentyp im DNS z.B. `SOA`, `A`, `AAAA`, `NS`, `MX`, `SRV`, `NAPTR`
  - DNSsec spez. RR `DNSKEY`, `RRSIG`, `DS`, `NSEC`, `NSEC3` (alt: `KEY`, `SIG`, `NXT`)
- Resource Record Set
  - Alle RRs mit gleichem Label und Typ

DNS Namensauflösung



Für Secure DNS:

- Zoneninhalte (z.B. A-Record von "www.example.net") werden signiert
- Stub Resolver (oder Recursive Resolver) kann die Signatur prüfen.

- Einführung
  - Warum DNSsec ? − Historie
  - Abkürzungen und Terminologie
  - DNS Namensauflösung − Ein Überblick
- Signierte Zonen
  - Schlüsselerzeugung / Signieren der Zone
  - Key- und Zone Signing Keys
  - Signieren des Parent / Secure Delegation
- Werkzeuge
  - Administrative Aufgaben
  - Beispiel: Zone Key Tool
- Secure Resolver
  - The big view: Chain of Trust
  - Resolver Konfiguration (Trusted-Keys)
  - Secure Root Testbed

Beispielzone

Filename: `zone.db`

```
$ORIGIN sec.example.net.

@   7200    IN SOA  ns1.example.net. hostmaster.example.net.  (
                        42      ; Serial No.
                        86400   ; refresh (24 hours)
                        7200    ; retry (2 hours)
                        1209600 ; expire (2 weeks)
                        3600    ; neg TTL (1 hour)

@       7200    IN NS       ns1.example.net.
        7200    IN NS       ns2.example.net.

@       7200    IN MX       10 mail.sec.example.net.
        7200    IN MX       20 backupmailer.ex.org.

mail    7200    IN A        1.2.3.4
        7200    IN AAAA     2001:0db8:0:123::25
```

DNSKEY − Schlüssel zu signierten Zonen

Kommando `dnssec-keygen` mit den folgenden Parametern:

- Schlüsselalgorithmus und Keylänge
  - DSA (Keysize 512 bis 1024 Bit)
  - RSAMD5 (Keysize 512 bis 4096 Bit)
  - RSASHA1 (Keysize 512 bis 4096 Bit)
- Key Namenstyp: ZONE
- Schlüsselname (Owner) = Domainname

```
$ dnssec-keygen -a RSASHA1 -b 512 -n ZONE sec.example.net
Ksec.example.net.+005+14417
                   ^   ^

Algorithmus ------+   |
Key ID ---------------+
```

Zwei Dateien:

```
-rw-r--r--  1 hoz hoz  125 Oct 11 10:31 Ksec.example.net.+005+14417.key
-rw-------  1 hoz hoz  549 Oct 11 10:31 Ksec.example.net.+005+14417.private
```

Einfügen des Keys in die Zone

- Der öffentliche Teil des Keys steht als RR in der Datei K*.key:

```
sec.example.net. IN DNSKEY 256 3 5 AQPUSMEKBKBsYO/xd...
                           ^   ^ ^ ^

 Flags: Bit8 == Zonenkey --+   | | |
 Protokoll (3 == DNS) ---------+ | |
 Algorithmus --------------------+ |
 Schlüsselmaterial (gekürzt) ------+
```

- Dateiname des Keys ändert sich bei Neugenerierung

```
$ cat Ksec.example.net.+00*.key > keys.db
```

- Einfügen der Keys in die Zone ($INCLUDE Anweisung)

```
$ cat zone.db
@   7200 IN  SOA  ns1.example.net. hostmaster.example.net. ....
         IN  NS          ns1.example.net.
         IN  NS          ns2.example.net.
$INCLUDE keys.db
    ....
```

- Erhöhen der Seriennummer! (Ab BIND 9.4 geht das mit dnssec-signzone)

RRSIG – Unterschriebene Resource Records

- Signieren der Zone durch `dnssec-signzone`

```
$ dnssec-signzone -e +864000 -N increment -o sec.example.net zone.db
zone.db.signed
```

  a. Sortieren der RR-Sets (RR-Set: Alle RR gleichen Typs eines Labels)
  b. Einfügen der NSEC Records
  c. Signieren jedes RR-Sets und Einfügen des Signatur Records (RRSIG)

```
$ cat zone.db.signed
    ...
sec.example.net.   7200 IN NS     ns1.example.net.
                   7200 IN NS     ns2.example.net.
                   7200 IN RRSIG  NS 1 2 7200 (
Sig. Lifetime                     20071011095802 20071021095802
Keytag+Name                       14417 sec.example.net.
Signaturdaten                     AK9adL3Ov7VkVLYoan/5CHUO..== )
```

- Nicht Vergessen: Vor Ablauf der Signatur erneut signieren!
  (d.h. nach obigem Beispiel z.B. alle 7 Tage)

RRSIG − Beispielzone

```
$ORIGIN sec.example.net.
@       7200  IN SOA     ns1.example.net. hostmaster.example.net.  28 43200 1800 1209600 3600
        7200  IN RRSIG   SOA 5 3 7200 20071031095802 20071021095802 14417 sec.example.net. lLp975...JbIjg+BQ==

        7200  IN NS      ns1.example.net.
        7200  IN NS      ns2.example.net.
        7200  IN RRSIG   NS 5 3 7200 20071031095802 20071021095802 14417 sec.example.net. UW5d03...hxVMDJ2g==

        7200  IN MX      10 mail.sec.example.net.
        7200  IN MX      20 backupmailer.ex.org.
        7200  IN RRSIG   MX 5 3 7200 20071031095802 20071021095802 14417 sec.example.net.  xyje/A...2RSkd8KA==

        7200  IN DNSKEY  256 3 5 (
                         BQEAAAABxtWpFzXTPqHZzPLLJzifEV7Hqrt4
                         wIu6BWRDFKORkdawLV8Bww==
                         ) ; key id = 14417
        7200  IN RRSIG   DNSKEY 5 2 7200 20071031095802 20071021095802 14417 sec.example.net. (
                         JMSXms...wq7CAHGkf3DbLxaPLJvjg9GwwxtRML
                         trXPBR...bm9jBV1FzXnw== )

        3600  IN NSEC    mail.sec.example.de. NS SOA MX NAPTR RRSIG NSEC DNSKEY
        3600  IN RRSIG   NSEC 5 3 3600 20071031095802 20071021095802 14417 sec.example.net.  dDvu...jZ0/pOQ==

mail    7200  IN A       1.2.3.4
        7200  IN RRSIG   A 5 4 7200 20071031095802 20071021095802 14417 sec.example.net. zLhUjl...GtzUrTd7==

        7200  IN AAAA    2001:0db8:0:123::25
        7200  IN RRSIG   AAAA 5 4 7200 20071031095802 20071021095802 14417 sec.example.net. zLdyXA...Had9g==

        3600  IN NSEC    sec.example.de. A AAAA RRSIG NSEC
        3600  IN RRSIG   NSEC 5 4 3600 20071031095802 20071021095802 14417 sec.example.net.  dDvu...jZ0/pOQ==
```

Key- und Zone Signing Keys

- Key Signing Keys (KSK)
  - Wird lediglich zum Signieren der Zonenkeys verwendet
  - Wenig genutzt (kleine Menge zu signierender Daten)
  - Große Schlüssellänge (DSA 1024, RSA 2048)
  - Lange Lebensdauer, d.h. selten geändert (> 1 Jahr)
  - Änderung des KSK muß kommuniziert werden!
  - KSK über ein Bit im Flagfeld gekennzeichnet (RFC3757)
- Zone Signing Keys (ZSK)
  - Wird zum Signieren der Zonendaten verwendet
  - Häufig genutzt (große Menge zu signierender Daten)
  - Kleine Schlüssellänge (RSA 512 Bit)
  - Kurze Lebensdauer (≈ Monat)
  - Änderung des Schlüssels muß nicht kommuniziert werden

KSK + ZSK (Konfiguration)

- Generieren des KSK (Option -f KSK)

```
$ dnssec-keygen -f KSK -n ZONE -a DSA -b 1024 sec.example.net
Ksec.example.net.+003+16004
```

- Generieren eines zweiten ZSK

```
$ dnssec-keygen -n ZONE -a RSASHA1 -b 512 sec.example.net
Ksec.example.net.+005+57764
```

- Einfügen der Keys in die Zone

```
$ cat Ksec.example.net.00[135]+*.key > keys.db
```

- Erhöhen der Seriennummer und Signieren

```
$ dnssec-signzone -e +864000 -N increment -o sec.example.net zone.db
zone.db.signed
```

- Neuladen

```
$ rndc reload sec.example.net
```

KSK + ZSK (Beispielzone)

```
$ORIGIN sec.example.net.
@       SOA     ns1.example.net. hostmaster.example.net.  29 43200 1800 1209600 3600
        RRSIG   SOA 5 3 7200 20071031143206 20071021143206 14417 sec.example.net. lLp9...bIjg+BQ==
        RRSIG   SOA 5 3 7200 20071031143206 20071021143206 57764 sec.example.net. AwEA...GHDIYkYJv9IYM=

        NS      ns1.example.net.
        NS      ns2.example.net.
        RRSIG   NS 5 3 7200 20071031143206 20071021143206 14417 sec.example.net. UW5d0...VMDJ2g==
        RRSIG   NS 5 3 7200 20071031143206 20071021143206 57764 sec.example.net. UW5d0...VMDJ2g==

        DNSKEY  257 3 3 (    AwEAAc1aS+ea+pmtcoZbQBjfP2aODgcTsyg0oGU1Ts/rdNWpU05TEmZP
                               w2KJByOqVl45FVuxRn8RnH5jO4Eirply4D7FVdD7E/PqgMQi9rWpqSm2
                                    ...
                               rGcBfHVLTLARyFgWXNHVYO0=          ) ; key id = 16004
        DNSKEY  256 3 5 (    BQEAAAABxtWpFzXTPqHZzPLLJzifEV7Hqrt4
                                    ...
                               wIu6BWRDFKORkdawLV8Bww==          ) ; key id = 14417
        DNSKEY  256 3 5 (    AwEAAcKGHDIqQR7m0Qg2lJTy+m6mWm+W4/T9
                               +QB27H8TlV0T3BSN8L9E4YWY8c1UkY7tHAzS
                               ts3egTZdIWAwYkPPYJv9IYM=          ) ; key id = 57764
        RRSIG   DNSKEY 3 3 7200 20071031143206 20071021143206 16004 sec.example.net. JMSXm...trXXnw==
        RRSIG   DNSKEY 5 3 7200 20071031143206 20071021143206 14417 sec.example.net. c6m/Y...KT8Yyw==
        RRSIG   DNSKEY 5 3 7200 20071031143206 20071021143206 57764 sec.example.net. c6m/Y...KT8Yyw==

www     A       1.2.3.5
        RRSIG   A 5 4 7200 20071031143206 20071021143206 14417 sec.example.net. zLdyX...K2sd9g==
        RRSIG   DNSKEY 5 3 7200 20071031143206 20071021143206 57764 sec.example.net. BFA2...7OzwQTA==
```

DS – Delegation Signer

- Delegation: Einfügen eines Verweises auf den KSK in der Parentzone
  `dnssec-signzone` erzeugt `dsset-` und `keyset-`Datei

- Die `keyset-`Datei enthält die DNSKEY-RR der **Key** Signing Keys

```
$ cat keyset-sec.example.net.
sec.example.net. 7200  IN DNSKEY 257 3 3 (
                       62uVBWg9spPDjXVaaXNaEwjLlNaKEqfwz4+A...
                       ) ; key id = 16004
```

- Die `dsset-`Datei enthält die **DS-RR**s als Verweis auf die KSKs

```
$ cat dsset-sec.example.net.
sec.example.net.  IN DS 16004 3 1 55FBEE63...
sec.example.net.  IN DS 16004 3 2 76A995910BDEA55521F2...
 Key Tag ---------------^     ^ ^ ^
 Algorithm Number ------------+ | |
 Digest Type (SHA1/SHA256) -----+ +-- Hash des DNSKEY
```

- Je nach Policy muß eine der beiden Dateien zum Parent übertragen werden

DS – Secure the Parent

- Der Parent muß seine Zone signieren!
- Wir brauchen Schlüsselmaterial für den Parent (KSK, ZSK, usw.)
- Signieren der Parent Zone

```
$ dnssec-signzone -g -o example.net zone.db
zone.db.signed
```

- Das Ergebnis:

```
$ORIGIN example.net.
sec      7200   IN NS     ns1.example.net.
         7200   IN NS     ns2.example.net.
         7200   IN DS     16004 3 1 55FBEE63...
         7200   IN DS     16004 3 2 76A995910BDEA55521F2...
         7200   IN RRSIG  DS 1 3 7200 20071031173508 (
                          20071021173508 65516 example.net.
                          dCzVu1NC7s/EB8e7Ynsl.... )
```

- Der Parent signiert nicht die Delegation (NS-Records)
- Lediglich der DS Record wird durch den Parent signiert!

- Einführung
  - Warum DNSsec ? − Historie
  - Abkürzungen und Terminologie
  - DNS Namensauflösung − Ein Überblick
- Signierte Zonen
  - Schlüsselerzeugung / Signieren der Zone
  - Key- und Zone Signing Keys
  - Signieren des Parent / Secure Delegation
- Werkzeuge
  - Administrative Aufgaben
  - Beispiel: Zone Key Tool
- Secure Resolver
  - The big view: Chain of Trust
  - Resolver Konfiguration (Trusted-Keys)
  - Secure Root Testbed

DNSsec − Administrative Aufgaben

- Regelmäßiges re-signing der Zone
  - Optimale Signatur Gültigkeit (Abwägung: Aufwand vers. Sicherheit)
  - Resigning Intervall bestimmen (z.B.: 7 Tage bei 10 Tage Siglifetime)
- Laufzeit des ZSK überwachen
  - Bei Ablauf Rollover durchführen
  - Pre- Publish Verfahren „fehleranfällig“
- Laufzeit des KSK überwachen
  - Double Signature Verfahren
  - Kommunikation mit Parent (Registry) notwendig
- Notfallkonzepte bereitstellen
  - KSK oder ZSK Komprimitierung
  - Ausfall des Signing Servers

Administrative Aufgaben: Key Rollover

- RFC4641 „DNSsec Operational Practices“ definiert zwei Verfahren
- ZSK Rollover (pre-publish key)
  1. Generiere einen zweiten ZSK
  2. Publiziere beide Schlüssel; Nutze nur den alten zum Signieren
  3. Warte mindestens: propagation time + TTL des DNSKEY−RR
  4. Signiere mit neuem Schlüssel; Publiziere nach wie vor beide
  5. Warte mindestens: propagation time + max TTL der alten Zone
  6. Entferne den alten Schlüssel aus der Zone
- KSK Rollover (double signature)
  1. Generiere einen zweiten KSK
  2. Benutze beide Schlüssel zum Key Signing
  3. Sende den neuen DS−Record zum Parent
  4. Warte bis DS in der Parentzone ist + TTL des alten DS−RR
  5. Entferne den alten Schlüssel

Zone Key Tool (ZKT)

- Zwei Programme für Schlüsselmanagement und zum Signieren

```
$ dnssec-zkt
$ dnssec-signer -N /etc/named.conf
```

- Einfache Konfigurationsdatei (Policyfile) (Auszug aus `dnssec.conf`)

```
#   zone specific timing values
ResignInterval: 1w      # (604800 seconds)
Sigvalidity:    10d     # (864000 seconds)
Max_TTL:        8h      # (28800 seconds)
Propagation:    5m      # (300 seconds)

#   signing key parameters
KSK_lifetime:   1y
KSK_algo:       RSASHA1 # (Algorithm ID 5)
KSK_bits:       1300
ZSK_lifetime:   30d     # (2592000 seconds)
ZSK_algo:       RSASHA1 # (Algorithm ID 5)
ZSK_bits:       512
```

- Vollautomatischer ZSK Schlüsseltausch (pre-publish key algorithm)
- Automatische Erhöhung der Seriennummer in der Zonendatei
  Unterstützt sequentielle Nummerierung, `YYYYmmDDxx` Format und Unixtime

ZKT – Konfigurationsbeispiel

- Anlegen eines separaten Verzeichnisses für jede Zone
  (Verzeichnisname == Domainname)

```
$ mkdir example.net.
$ cd example.net.
```

- Einfügen einer Include Anweisung in die Zonendatei (`zone.db`)

```
$INCLUDE dnskey.db        ;include the DNSKEY records
```

- Spezielle Formatierung des SOA Records
  (Nur notwendig falls nicht das Unixtime Serialformat verwendet wird)

```
$  head -15 zone.db
$TTL 7200
;    Be sure that the serial number below is left
;    justified in a field of at least 10 chars !!
;                   0123456789;
@    IN SOA   ns1.example.net. hostmaster.example.net.  (
                   63         ; Serial
                   43200   ; Refresh
                   1800    ; Retry
                   2W      ; Expire
                   3600 )  ; Minimum (NegTTL)
```

ZKT − Konfigurationsbeispiel(2)

- Anlegen einer (leeren) Datei `zone.db.signed`

```
$ touch zone.db.signed
$ ls -l
-rw-r-----   1 dnsop  dnsop   916 2007-10-21 17:54 zone.db
-rw-r--r--   1 dnsop  dnsop     0 2007-10-21 17:55 zone.db.signed
```

- Signieren der Zone

```
$ dnssec-signer -v -o example.net.
parsing zone "example.net." in dir "."
        No active KSK found: generate new one
        No active ZSK found: generate new one
        Re-signing necessary: Modified keys
        Writing key file "./dnskey.db"
        Incrementing serial number in file "./zone.db"
        Signing zone "example.net."
$ ls -l
-rw-r--r--   1 dnsop  dnsop   581 2007-10-21 17:55 Kexample.net.+005+18710.key
-rw-------   1 dnsop  dnsop   688 2007-10-21 17:55 Kexample.net.+005+18710.private
-rw-r--r--   1 dnsop  dnsop   121 2007-10-21 17:55 Kexample.net.+005+57705.key
-rw-------   1 dnsop  dnsop   545 2007-10-21 17:55 Kexample.net.+005+57705.private
-rw-r--r--   1 dnsop  dnsop  1136 2007-10-21 17:55 dnskey.db
-rw-r--r--   1 dnsop  dnsop    71 2007-10-21 17:55 dsset-example.net.
-rw-r--r--   1 dnsop  dnsop   702 2007-10-21 17:55 keyset-example.net.
-rw-r-----   1 dnsop  dnsop   916 2007-10-21 17:55 zone.db
-rw-r--r--   1 dnsop  dnsop  4080 2007-10-21 17:55 zone.db.signed
```

ZKT − Konfigurationsbeispiel(3)

- Zeige den aktuellen Status der Schlüssel an

```
$  dnssec-zkt -a .
Keyname              Tag Typ Sta Algorit Generation Time            Age
example.net.     18710 KSK act RSASHA1 Oct 21 2007 18:08:24     13m42s
example.net.     57705 ZSK act RSASHA1 Oct 21 2007 18:08:24     13m42s
```

- Anpassen des Dateinamens in `named.conf`

```
zone "example.net." in {
        type master;
        file "example.net./zone.db.signed";
};
```

- Erzwinge ein re-signing und reload der Zone

```
$  dnssec-signer -f -r -v -N named.conf
parsing zone "example.net." in dir "./."
        Re-signing necessary: Option -f
        Incrementing serial number in file "././zone.db"
        Signing zone "example.net."
        Reload zone "example.net."
```

- Kontrolle des Logfile `/var/log/named`

21-Oct-2007 18:10:43.198 general: info: zone example.net/IN: loaded serial 65 **(signed)**

ZKT − Konfigurationsbeispiel(4)

- Regelmäßiges Neusignieren der Zone
  Aufruf von `dnssec-signer` einmal täglich (Abhängig von Resigning Intervall)
- `cron` is your friend

```
$ crontab -l
21  6 * * * /home/dnsop/dnssec-cron 2>&1 | logger -t dnssec-cron -p daemon.info
21 18 * * * /home/dnsop/dnssec-cron 2>&1 | logger -t dnssec-cron -p daemon.info
```

- Das `dnssec-cron` Script:

```
echo "current zone signing keys"
/home/dnsop/bin/dnssec-zkt -z
echo "dnssec re-signing process started"
/home/dnsop/bin/dnssec-signer -v -v -r -N /var/named/named.conf
```

- Erzeugen der `trusted-keys`-Section für die Resolver Konfiguration

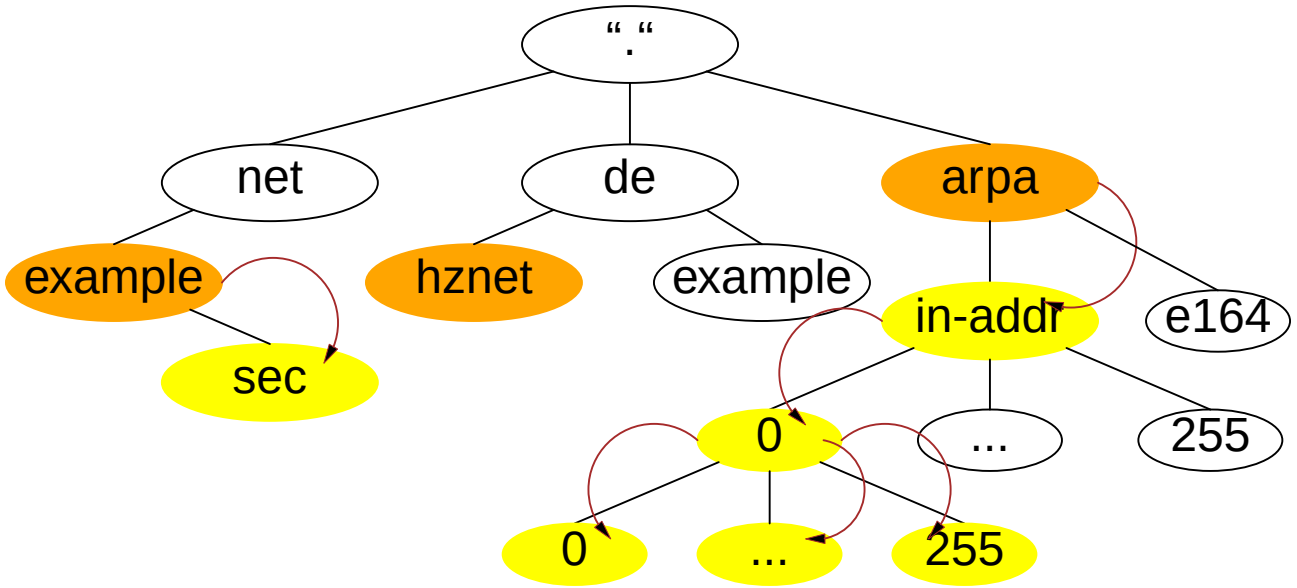
```
$ dnssec-zkt -T -l example.net.
trusted-keys {
"example.net."   257 3 5 "CJEUcyN1ES5bAnBI4O+m7nLhbmTfxVtF31O4agNVe+6Hu8kZ8EKzm+/U
                          +qh2NXv6+UgowadnPlfHHwLzpfNP4aZXfXa2qog1P5dp7POUquW6zn25
                                    ...
                          Wdlf/F/2lJh2LF4bU616EyOeRichLvlBXn15nkkLr4usbPitr68DrVas
                          o6bci4LJlPJBkHVS/3MtBo0lSY3XvoiBJtgp" ; # key id = 18710
};
```

ZKT − Key Rollover

- Automatischer Rollover des Zone Signing Keys
  - Initiierung vor Ablauf der ZSK-Lifetime
  - Pre-Publish Algorithmus
  - Voraussetzung: Regelmäßiger Aufruf von `dnssec-signer`
- Semiautomatischer Rollover des Key Signing Keys
  - Manuell initiiert `dnssec-zkt --ksk-rollover`
  - Ablauf in drei Schritten
    1. `dnssec-zkt --ksk-newkey do.ma.in.`
       Erzeugt neuen KSK und verwendet diesen **zusätzlich** bei der Signierung der DNSKEYs
    2. `dnssec-zkt --ksk-publish do.ma.in.`
       Erzeugt eine `parent-do.ma.in` Datei mit dem **neuen** KSK
       Automatisches „Senden“ zum Parent im hierarchischen Modus
    3. `dnssec-zkt --ksk-delkey do.ma.in.`
       Löschen (umbenennen) des alten KSK aus der Zone

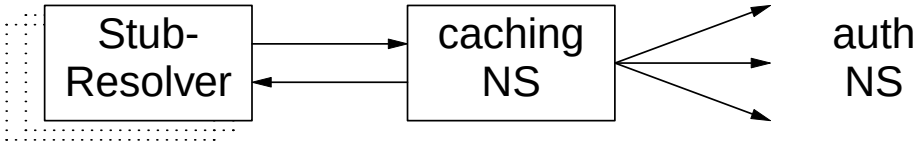
- Einführung
  - Warum DNSsec ? − Historie
  - Abkürzungen und Terminologie
  - DNS Namensauflösung − Ein Überblick
- Signierte Zonen
  - Schlüsselerzeugung / Signieren der Zone
  - Key- und Zone Signing Keys
  - Signieren des Parent / Secure Delegation
- Werkzeuge
  - Administrative Aufgaben
  - Beispiel: Zone Key Tool
- Secure Resolver
  - The big view: Chain of Trust
  - Resolver Konfiguration (Trusted-Keys)
  - Secure Root Testbed

Chain of Trust



- Chain of Trust durch DS Records
- Ohne „Signed Root“ viele „Trust Anchor“
- Trust-Anchor müssen beim Validator hinterlegt werden

DNSsec Validator



Zwei Modi:

a. Signaturprüfung durch den Caching NS (Resolver)
   - EDNS0: do-Flag in der Anfrage setzen
   - EDNS0: UDP-Size 4096
   - In der Antwort sollte AD-Bit gesetzt sein
     (verified secure/insecure)

b. Eigenprüfung der Signatur
   - Stub-Resolver benötigt Trust-Anchor (SEP)
   - Zusätzlich bei der Anfrage das CD-Flag setzen
   - Die Antwort enthält auch Authority Section
   - AD-Bit nicht gesetzt

dig als Stub-Resolver

```
$ dig @secResolver +multi +dnssec www.sec.example.net
; <<>> DiG 9.4.0b3 <<>> @secResolver +multi +dnssec www.sec.example.net
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42021
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 4, AUTHORITY: 5, ADDITIONAL: 11

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;www.sec.example.net.                IN A

;; ANSWER SECTION:
www.sec.example.net.     5147 IN A 1.2.3.4
www.sec.example.net.     5147 IN RRSIG A 1 4 7200 20071031143206 (
                              20071021143206 14417 sec.example.net.
                              EZ0P5FVLaaRYx09Gh5VWVJzySt9CTPDhRgwAE522+L93
                              27XecpZQwsKileKFdoExPQqQAWJJo4c9vUIZ+3tSBw== )
www.sec.example.net.     5147 IN RRSIG A 1 4 7200 20071031143206 (
                              20071021143206 57764 sec.example.net.
                              Ju5aWfSFGHppl+spF4/PVmB6vOZ/LBJQJqjGF/Du/tyS
                              gNvUdsGkNn0EN2hxp8Z6FByTOKrV1w4SQZufBs0EVw== )
;; Query time: 107 msec
;; SERVER: 1.2.3.105#53(secResolver)
;; WHEN: Sun Oct 21 19:37:51 2007
;; MSG SIZE  rcvd: 2458
```

Secure Resolver (Caching NS)

- BIND 9.4.1-P1 oder neuer benutzen
  Achtung: bind 9.4.2rc1 setzt AD-Bit nur nach DO-Query

- Secure DNS in `named.conf` einschalten

```
options {
    recursion yes;
    dnssec-enable yes;
    dnssec-validation yes;  /* required since BIND 9.4.0 */
};
```

- Trust Anchor in der „trusted-keys“-Section hinterlegen

```
trusted-keys {
    "example.net." 257 3 5 "AQPUSMEKBKBsYO/xdnL/j..."
};
```

- Wie kommt man an die Trust-Anchor ?

Trust Anchor Konfiguration

- Idealerweise wird lediglich ein TA (root) benötigt
  - IANA DNSsec testbed
    https://ns.iana.org/dnssec/status.html
  - Hintdatei im Resolver auf ns.iana.org zeigen lassen

```
$ grep "type hint" named.conf
zone "." in { type hint; file "signed-root.hint"; };

$ cat signed-root.hint
.                       518400  IN      NS      ns.iana.org.
ns.iana.org.            3600000 IN      A       208.77.188.32
```

  - Trusted Key konfigurieren

```
trusted-keys {
"." 257 3 5 "AwEAAcRgtTT1Szdg7y9/wQOYjvDJx3MkKH33iaRdjp24
                                ...
            AZsT4Vgo6iDOX31IEq5u0RefFrHAjFMuRF5n65c="; # key id = 14777
"." 257 3 5 "AwEAAbWMiPoQlFp+snq84lbEPx2kPgessP91ieS+jeab
                                ...
            0PNPWQHfPWp045wUAqrRagTbRs7sWw/fpKgC5I0="; # key id = 4183
};
```

Trust Anchor Konfiguration (2)

- Über „Delegated Verification“ (DLV)
  - IKS Jena
    https://www.iks-jena.de/leistungen/dnssec.php
  - Internet Systems Consortium (ISC)
    https://secure.isc.org/index.pl?/ops/dlv/
  - Achtung: Skalierungs- und Privacy Problem
- Download der Trust Anchor von spez. Webseiten
  - SecSpider
    http://secspider.cs.ucla.edu/ (neu: http://secspider.cs.ucla.edu/2.0)
  - RIPE
    https://www.ripe.net/projects/disi/keys/
  - .SE
    http://dnssec.nic.se/
  - Globales „Trust Anchor Repository“, gepflegt bei RIPE?
    Aktuell auf dem RIPE Meeting (letzten Mittwoch) diskutiert
    http://www.ripe.net/ripe/meetings/ripe-55/presentations/reid-taskforce-report.pdf

Zusammenfassung

- Grundlegende Mittel zum Signieren einer Zone vorhanden
  BIND + Tools (ZKT, dnssec-tools, Key management tools)
- Erste signierte TLDs verfügbar
  - `.se`, `.bg`, `.br`, `.e164.arpa` (ab 26. Nov 07), `arpa` (coming soon)
  - Alle RIPE Zonen (inkl. reverse `.in-addr.arpa`, `.ip6.arpa`)
- Applikationen die DNSsec unterstützen
  - OpenSSH (s.a.: „SSH Fingerprints in the Domain Name System“)
  - dnssec-tools.org: Sendmail/Postfix/libspf, Thunderbird, Firefox
- Was fehlt:
  - Standards, Prozesse und Tools für DS Registrierung
  - Secure (Stub)Resolver (Bibliotheken bereits in Entwicklung)
  - Werkzeuge zum Resolver Management (TA−Verwaltung)
  - Mehr secure TLDs! (`.de`, `.arpa`, `.com`, `.net`, `.org`, `.eu`)

Referenzen

Olaf Kolkman, NLnetlabs, Ripe NCC
„DNSSEC Howto Version 1.8.2“
(*http://www.nlnetlabs.nl/dnssec_howto/*")

Internet Systems Consortium
BIND v9 Administrator Reference Manual
(*http://www.isc.org/sw/bind/arm94/*)

RFCs 4033 (DNS Security Introduction and Requirements)
4034 (Resource Records for the DNS Security Extentions)
4035 (Protocol Modifications for the DNS Security Extensions)
4641 (DNSSEC Operational Practices)
5011 (Automated Updates of DNS Security (DNSSEC) Trust Anchors)

Links http://www.dnssec.net
http://secspider.cs.ucla.edu/secspider/
http://www.iks-jena.de/leistungen/dnssec/
http://www.dnssec-deployment.org
http://www.hznet.de/zkt/

Noch Fragen ?

Jetzt, oder im Anschluß ...

oder

Fragen sie nach einem individuellen intensiv Workshop zu...

- DNS
- ENUM
- IPv4/IPv6
- SIP
- DNSsec
- DHCP
- Mail (DKIM)
- VoIPsec
- DynamicDNS
- Routing
- Security
- ...

Noch Fragen ?

Jetzt, oder im Anschluß ...

oder

Fragen sie nach einem individuellen intensiv Workshop zu...

- DNS
- ENUM
- IPv4/IPv6
- SIP
- DNSsec
- DHCP
- Mail (DKIM)
- VoIPsec
- DynamicDNS
- Routing
- Security
- ...

Herzlichen Dank für Ihre Aufmerksamkeit!

CONTENTS