Secure DNS

Eine Einführung

Secure Linux Administration Conference

Berlin
07./08. Dezember 2006

Holger.Zuleger@hznet.de

Agenda

- Einführung
- Signierte Zonen
  - Schlüsselerzeugung
  - Signieren der Zone
  - Secure Resolver
- Chain of Trust
  - Key- und Zone Signing Keys
  - Signieren des Parent
  - Secure Delegation
- DNSsec und Privacy: NSEC und die Folgen
- Key Rollover
- Werkzeuge
  - Überblick
  - Beispiel: Zone Key Tool

DNS/DNSsec − Historie

Siehe: http://www.nlnetlabs.nl/dnssec/history.html

1983	„Erfindung“ des DNS durch Paul Mockapetris
1986	Formale Definition: RFC 1034 und 1035
1990	Steven Bellovin beschreibt Angriffsszenarien gegen DNS „Using the Domain Name System for System Break-ins“
1995	Veröffentlichung des Bellovin Paper und Beginn der Entwicklung an DNSSEC
1999	RFC 2535 wird veröffentlicht, findet allerdings keine Verbreitung
2001	Entwicklung von Verfahren für einfacheres Key Handling (DS)
2002	Neufassung des DNSsec Standard RFC2535bis
2005	Verabschiedung RFC4033, RFC4034, RFC4035
2006	RFC 4641 DNSSEC Operational Practices

DNSsec: Was, Wie, Warum?

- Secure DNS addressiert unterschiedliche Problemstellungen:
  a. Authentisierte Kommunikation
     - Zonentransfer (Slave − Master)
     - Dynamic Update (DHCP-Server − Master)
     - Dynamic Update (Client − Master)
     - Querys (Stub-Resolver − Caching Resolver)
  b. Authentisierte Zoneninhalte
     Kryptographische Signatur über den Zoneninhalt
- Implementierung: ab bind-9.3.x, NSD 2.1.x
- Bedeutung des DNS nimmt zu
  - Phishing/Pharming
  - Immer mehr Dienste benötigen DNS
    (Anti SPAM (DKIM/SPF), ENUM, SIP, SSH-Fingerprints, IPSECKEY)
  - Immer mehr Dienste benötigen mehrstufige DNS Auflösung
    (MX, SRV, NAPTR)

DNSsec − Resource Records

- TSIG (RFC2848)
  Transaktions Signaturen (Hashed MD5)
  Authentisierte Zonentransfers sowie sig. Querys und Updates

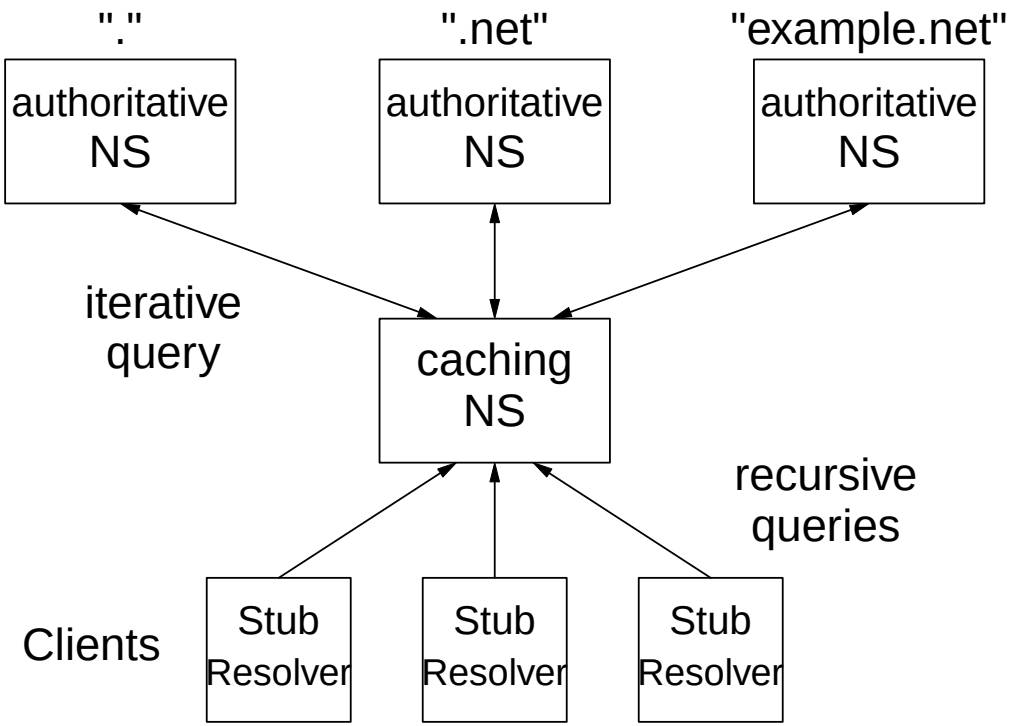
- TKEY (RFC2930)
  Verfahren zur Aushandlung von symmetrischen Keys für TSIG
  (Diffie-Hellman, Sig(0), GSSAPI)

- SIG(0) (RFC2931)
  Transaktions Signaturen (Public-Key: RSA-MD5, RSA-SHA1, DSA)
  Authentisierte dynamische Updates und TKEY Request

- RRSIG, DNSKEY, NSEC, NSEC3 (RFC4035, RFC3845)
  Signierte Resource Records (Public Key Verfahren)
  Ehemals: SIG, KEY, NXT

- DS (RFC3658)
  Delegation der Vertrauensbeziehung

Authentisierte Namensauflösung



- Host zu Host Authentisierung hierbei nicht praktikabel
- Stattdessen: Signatur des Zoneninhalts (authenticated data origin)

Authentisierte Zoneninhalte

Authoritative NS (Master)

- Generieren eines asymetrischen Keys (Zonenkey)
- Einfügen des (öffentlichen) Keys in die Zone
- Erhöhen der Seriennummer im SOA Record
- Signieren der Zone (Offline)
- Reload der Zone

Resolving NS

- DNSsec einschalten
- Öffentlicher Zonenkey als „Secure Entry Point“ (SEP) hinterlegen

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 Nov 11 10:31 Ksec.example.net.+005+14417.key
-rw-------  1 hoz hoz  549 Nov 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 !

RRSIG – Unterschriebene Resource Records

- Signieren der Zone durch `dnssec-signzone`

```
$ dnssec-signzone -e +864000 -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                    20061121105802 20061111105802
Keytag+Name                      14417 sec.example.net.
Signaturdaten                    AK9adL3Ov7VkVLYoan/5CHUO..== )
```

- Vor Ablauf der Signatur erneut signieren!

RRSIG − Beispielzone

```
$ORIGIN sec.example.net.
@       3600  IN SOA     ns1.example.net. hostmaster.example.net.  28 43200 1800 1209600 7200
        3600  IN RRSIG   SOA 5 3 3600 20061121105802 20061111105802 14417 sec.example.net. lLp975...JbIjg+BQ==

        3600  IN NS      ns1.example.net.
        3600  IN NS      ns2.example.net.
        3600  IN RRSIG   NS 5 3 3600 20061121105802 20061111105802 14417 sec.example.net. UW5d03...hxVMDJ2g==

        3600  IN MX      10 mailer.sec.example.net.
        3600  IN MX      20 backupmailer.ex.org.
        3600  IN RRSIG   MX 5 3 3600 20061121105802 20061111105802 14417 sec.example.net.  xyje/A...2RSkd8KA==

        3600  IN DNSKEY  256 3 5 (
                         BQEAAAABxtWpFzXTPqHZzPLLJzifEV7Hqrt4
                         wIu6BWRDFKORkdawLV8Bww==
                         ) ; key id = 14417
        3600  IN RRSIG   DNSKEY 5 2 3600 20061121105802 20061111105802 14417 sec.example.net. (
                         JMSXms...wq7CAHGkf3DbLxaPLJvjg9GwwxtRML
                         trXPBR...bm9jBV1FzXnw== )

        7200  IN NSEC    mailer.sec.example.de. NS SOA MX NAPTR RRSIG NSEC DNSKEY
        7200  IN RRSIG   NSEC 5 3 7200 20061121105802 20061111105802 14417 sec.example.net.  dDvu...jZ0/pOQ==

mailer  3600  IN A       1.2.3.4
        3600  IN RRSIG   A 5 4 3600 20061121105802 20061111105802 14417 sec.example.net. zLhUjl...GtzUrTd7==

        3600  IN AAAA    2001:0db8:0:123::25
        3600  IN RRSIG   AAAA 5 4 3600 20061121105802 20061111105802 14417 sec.example.net. zLdyXA...Had9g==

        7200  IN NSEC    sec.example.de. A AAAA RRSIG NSEC
        7200  IN RRSIG   NSEC 5 4 7200 20061121105802 20061111105802 14417 sec.example.net.  dDvu...jZ0/pOQ==
```

Reload der Zone

- Name des Zonenfiles in der `named.conf` eintragen:

```
zone "sec.example.net" in {
        type master;
        file "sec.example.net/zone.db.signed";
        allow-transfer { key ns1.example.net-ns2.example.net.;};
        notify yes;
};
```

- Zone neu laden

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

- Meldungen kontrollieren

```
$ tail -f /var/log/named
11-Nov-2006 13:59:43.198 general: info: zone sec.example.net/IN: \
                                        loaded serial 28 (signed)
```

Secure Resolver (Caching NS)

- BIND 9.4.x benutzen
- Secure DNS in `named.conf` einschalten

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

- Secure Entry Point in der „trusted-keys“-Section hinterlegen

```
trusted-keys {
    "sec.example.net." 256 3 5 "AQPUSMEKBKBsYO/xdnL/j..."
};
```

- Wie ?

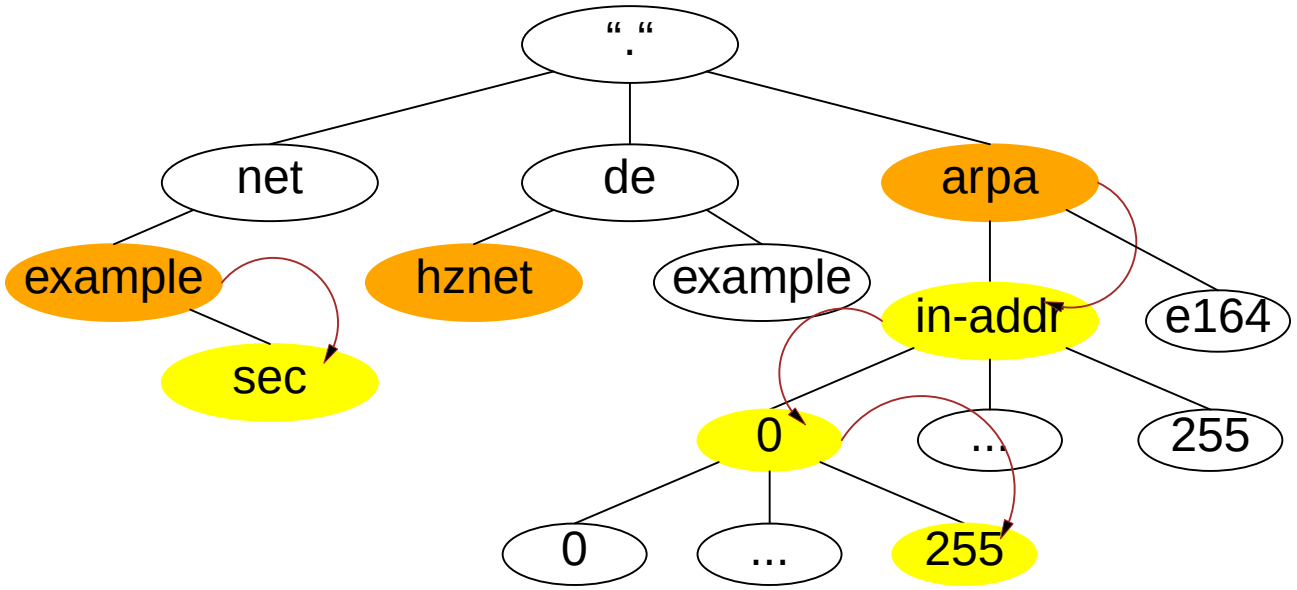
```
$ { echo "trusted-keys {";
    cat Ksec.example.net.+005+*.key  |
    sed -n 's/^\([a-zA-Z.]*\) IN DNSKEY \([0-9]*\) \([0-9]\) \
            \([0-9]\) \(.*\)/"\1\" \2 \3 \4 "\5"/p'
    echo "};"
} >> named.conf
```

Oder besser: http://www.hznet.de/dns/create-trusted-key-section

Nächste Schritte

- Verteilung der „Secure Entry Points“
  Wie? Skalierbarkeit?
- Jeder (secure) Resolver benötigt den öffentlichen Zonenkey von **allen** signierten Zonen!
  - Delegation Signer d.h. Aufbau einer Hierarchie
  - Delegated Verification (Sichere Inseln)
- Änderung des Zone Signing Key erfordert Anpassung des Resolver
  Lösung: Key Signing Keys (KSK)
- Wie kann eine negative Antwort signiert werden ?
  - NSEC Resource Records
  - NSEC3 RR (hashed NSEC)
  - Online signing
- Key Rollover

Chain of Trust / Secure Entry Points



- Chain of Trust durch DS Records
- Wenige(r) „Secure Entry Points“
- Delegated Verification weicht hierarchisches Trust Modell auf (Experimental)
- Minimaler Schlüsselaustausch durch Key Signing Keys

KSK + ZSK

- Key Signing Keys (SEP)
  - 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
  - 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!
- Signieren + Neuladen

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

- KSK in der trusted-key Section des Resolver eintragen!

KSK + ZSK (Beispielzone)

```
$ORIGIN sec.example.net.
@       SOA     ns1.example.net. hostmaster.example.net.  29 43200 1800 1209600 7200
        RRSIG   SOA 5 3 3600 20061121143206 20061111143206 14417 sec.example.net. lLp9...bIjg+BQ==
        RRSIG   SOA 5 3 3600 20061121143206 20061111143206 57764 sec.example.net. AwEA...GHDIYkYJv9IYM=

        NS      ns1.example.net.
        NS      ns2.example.net.
        RRSIG   NS 5 3 3600 20061121143206 20061111143206 14417 sec.example.net. UW5d0...VMDJ2g==
        RRSIG   NS 5 3 3600 20061121143206 20061111143206 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 3600 20061121143206 20061111143206 16004 sec.example.net. JMSXm...trXXnw==
        RRSIG   DNSKEY 5 3 3600 20061121143206 20061111143206 14417 sec.example.net. c6m/Y...KT8Yyw==
        RRSIG   DNSKEY 5 3 3600 20061121143206 20061111143206 57764 sec.example.net. c6m/Y...KT8Yyw==

www     A       1.2.3.5
        RRSIG   A 5 4 3600 20061121143206 20061111143206 14417 sec.example.net. zLdyX...K2sd9g==
        RRSIG   DNSKEY 5 3 3600 20061121143206 20061111143206 57764 sec.example.net. BFA2...7OzwQTA==
```

DS – Delegation Signer

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

- Die `keyset-`Datei enthält die DNSKEY-RR der Key Signing Keys
  Diese werden in der secure Zone (!) eingefügt

```
$ 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-RRs als Verweis auf die KSKs
  Diese werden in der Parent-Zone (!) eingefügt.

```
$ cat dsset-sec.example.net.
sec.example.net.  IN DS 16004 3 1 55FBEE63...
 Key Tag ---------------^     ^ ^ ^
 Algorithm Number ------------+ | |
 Digest Type (SHA1) ------------+ +-- 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:

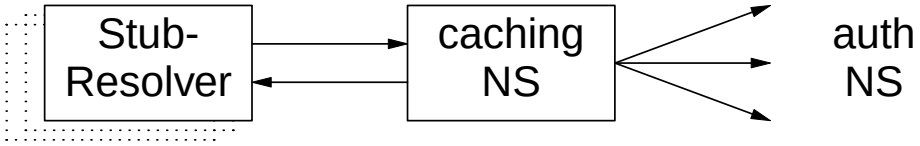
```
$ORIGIN example.net.
sec      7200   IN NS    ns1.example.net.
         7200   IN NS    ns2.example.net.
         7200   iN DS    16004 3 1 (55FBEE63... )
         7200   iN RRSIG DS 1 3 7200 20061211173208 (
                         20061111173208 65516 example.net.
                         dCzVu1NC7s/EB8e7Ynsl.... )
```

- Der Parent signiert nicht die Delegation (NS-Records)

  Lediglich der DS Record wird durch den Parent signiert!

- Resolver benötigt den SEP des Parent in der trusted-key Section

Stub-Resolver / Clients



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

Stub-Resolver (dig)

```
$ 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 20061121143206 (
                              20061111143206 14417 sec.example.net.
                              EZ0P5FVLaaRYx09Gh5VWVJzySt9CTPDhRgwAE522+L93
                              27XecpZQwsKileKFdoExPQqQAWJJo4c9vUIZ+3tSBw== )
www.sec.example.net.     5147 IN RRSIG A 1 4 7200 20061121143206 (
                              20061111143206 57764 sec.example.net.
                              Ju5aWfSFGHppl+spF4/PVmB6vOZ/LBJQJqjGF/Du/tyS
                              gNvUdsGkNn0EN2hxp8Z6FByTOKrV1w4SQZufBs0EVw== )
;; Query time: 107 msec
;; SERVER: 1.2.3.105#53(secResolver)
;; WHEN: Sat Nov 11 15:37:51 2006
;; MSG SIZE  rcvd: 2458
```

Signiertes Nichts: NSEC ...

Wie kann eine negative Antwort (offline) signiert werden?

- NSEC (früher NXT) Pseudorecord (Next SECure Record)
  - Alle Records sortieren
  - Bei jedem Label ein NSEC als Zeiger auf das nächste Label einfügen
  - Signieren der Zone

```
example.net.      SOA  ns1.example.net. ...
                  NS   ns1.example.net.
                  NS   ns2.example.net.
                  NSEC a.example.net. NS SOA RRSIG NSEC DNSKEY

a.example.net.    A    1.2.3.4
                  NSEC  b.example.net. A RRSIG NSEC

b.example.net.    A    1.2.3.5
                  NSEC example.net. A RRSIG NSEC
```

- Funktioniert auch mit Wildcards

... und die Folgen

- Ermöglicht einfaches Auslesen aller Labels einer Zone (Zonewalk)

  Prinzipielle Arbeitsweise:

```
$ dig +noall +answer nsec example.net
example.net.    7171  IN NSEC    a.example.net. A RRSIG NSEC

$ dig +noall +answer nsec a.example.net
a.example.net.  7171  IN NSEC    b.example.net. A RRSIG NSEC

$ dig +noall +answer nsec b.example.net
b.example.net.  7171  IN NSEC    example.net. A RRSIG NSEC
```

  Siehe auch: DNSSEC Walker (http://josefsson.org/walker/)

- Alternativen zu NSEC werden zur Zeit noch diskutiert
  - DNSSEC Hashed Authenticated Denial of Existence
    draft-ietf-dnsext-nsec3-08.txt
  - Minimally Covering NSEC Records and DNSSEC On-line Signing
    RFC4470

Schlüsseltausch (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

DNSsec − Administrative Aufgaben

- Regelmäßiges re-signing der Zone
  - Optimale Signatur Gültigkeit
  - Resigning Intervall bestimmen
- 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

DNSsec Werkzeuge

KROd − Key Rollover Daemon (www.idsa.prd.fr/index.php?page=kro&lang=en)
- — Vollautomatischer KSK und ZSK Schlüsseltausch
- — C−Programm als Frontend zu den BIND tools

- DNSSEC Key Maintenance Tools (www.ripe.net/disi/code.html)
  - — Halbautomatischer KSK und ZSK Schlüsseltausch
  - — Benutzt sicheren Bereich zur Speicherung der privaten Schlüssel
  - — Perl basierend
- DNSsec Tools (www.dnssec-tools.org)
  - — Perl basierter Wrapper um die BIND tools
  - — Inkl. DNSsec Resolver Bibliotheken (Sendmail, Mozilla, Thunderbird)
- Zone Key Tool (www.hznet.de/zkt/)
  - — Vollautomatisches Re-Signing der Zone und ZSK Schlüsseltausch
  - — C basiertes Frontend zu den BIND tools (Standard BIND Konfiguration)
  - — Entwickelt für kleine bis mittlere Anzahl von Zonen

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:   0
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 und `YYYYmmDDxx` Format

ZKT − Konfigurationsbeispiel

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

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

- Format der Zonendatei (Standardname: `zone.db`)

```
$  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
                 7200 )   ; Minimum

          IN  NS  ns1.example.net.
          IN  NS  ns2.example.net.

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

ZKT − Konfigurationsbeispiel(2)

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

```
$ touch zone.db.signed
$ ls -l
-rw-r-----   1 dnsop  dnsop   916 2006-11-11 15:54 zone.db
-rw-r--r--   1 dnsop  dnsop     0 2006-11-11 15:55 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 (64) in file "./zone.db"
        Signing zone "example.net."
$ ls -l
-rw-r--r--   1 dnsop  dnsop   581 2006-11-11 15:55 Kexample.net.+005+18710.key
-rw-------   1 dnsop  dnsop   688 2006-11-11 15:55 Kexample.net.+005+18710.private
-rw-r--r--   1 dnsop  dnsop   121 2006-11-11 15:55 Kexample.net.+005+57705.key
-rw-------   1 dnsop  dnsop   545 2006-11-11 15:55 Kexample.net.+005+57705.private
-rw-r--r--   1 dnsop  dnsop  1136 2006-11-11 15:55 dnskey.db
-rw-r--r--   1 dnsop  dnsop    71 2006-11-11 15:55 dsset-example.net.
-rw-r--r--   1 dnsop  dnsop   702 2006-11-11 15:55 keyset-example.net.
-rw-r-----   1 dnsop  dnsop   916 2006-11-11 15:55 zone.db
-rw-r--r--   1 dnsop  dnsop  4080 2006-11-11 15:55 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 Nov 11 2006 16:08:24     13m42s
example.net.     57705 ZSK act RSASHA1 Nov 11 2006 16:08:24     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
        Writing key file "././dnskey.db"
        Incrementing serial number (65) in file "././zone.db"
        Signing zone "example.net."
        Reload zone "example.net."
```

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

11-Nov-2006 16:10:43.198 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 ist einfach

```
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
};
```

Zusammenfassung

- BIND-Tools bieten grundlegende Mittel zum Signieren einer Zone
- Zusätzliche Werkzeuge stehen bereit:
  - Für Schlüsselmanagement
  - Für Automatisierung des Signing Prozesses
- Erste signierte Zonen verfügbar
  - `.se`
  - Alle RIPE Reverse Zonen (`.in-addr.arpa`, `.ip6.arpa`)
  - Zur Zeit ca. 600 secured Zones
- Was fehlt:
  - Standards, Prozesse und Tools für DS Registrierung
  - Secure Resolver (Einige Bibliotheken bereits in Entwicklung)
  - Werkzeuge zum Resolver Management (SEP−Verwaltung)
  - Mehr secure TLDs! (`.de`, `.arpa`, `.com`, `.net`, `.org`, `.eu`)

Referenzen

Olaf Kolkman, Ripe-NCC DISI
„DNSSEC Howto Version 1.5“
(*http://www.ripe.net/disi/dnssec_howto/dnssec_howto.pdf*)

Nominum
BIND v9 Administrator Reference Manual
(*http://www.isc.org/sw/bind/arm93/Bv9ARM.pdf*)

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)

Drafts DNSSEC Hashed Authenticated Denial of Existence
draft-ietf-dnsext-nsec3-08.txt

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/

Fragen ?

Fragen ?

http://www.hznet.de/dns/dnssec-slac061208.pdf

Fragen ?

http://www.hznet.de/dns/dnssec-slac061208.pdf

Herzlichen Dank für Ihre Aufmerksamkeit!

CONTENTS