DNSSEC

or

How to secure your (reverse) zone

DE-CIX 3rd technical workshop
16. Sep 2005

Holger.Zuleger@hznet.de
Agenda

- Secure DNS
  - Why do we need it?
  - What is that?

- Overview
  - Signing
  - Key generation
  - Chain of Trust
  - Secure Resolver

- DNSsec Practice

- Key rollover

- DNSsec Tools
Why DNSsec?

- DNS (RFC1034, RFC1035) is bad designed! ('86, first vulnerability '90)
Why DNSsec?

- DNS (RFC1034, RFC1035) is bad designed! ('86, first vulnerability '90)
Why DNSsec?

- DNS (RFC1034, RFC1035) is bad designed! ('86, first vulnerability '90)

Data storage and maintenance

Query process

- Impersonating Master
- Altered zone data
- Cache pollution
- Data spoofing
- Cache impersonation (IP-Spoofing)

Unauthorized updates

dynamic updates

Slave NS(s)

Master NS

Caching NS

Resolver

Resolver

zone.db
Why DNSsec?

- DNS (RFC1034, RFC1035) is bad designed! ('86, first vulnerability '90)

Data storage and maintenance
- Impersonating Master
- Altered zone data
- Unauthenticated updates
- Server protection

Query process
- Dynamic updates
- Slave NS(s)
- Caching NS
- Resolver
- Data spoofing
- Cache impersonation (IP-Spoofing)
- Data protection
Why DNSsec?

- DNS (RFC1034, RFC1035) is badly designed! ('86, first vulnerability '90)

Data storage and maintenance

- Impersonating Master
- Altered zone data
- Unauthorized updates

Dynamic updates

Query process

- Cache pollution
- Cache impersonation (IP-Spoofing)

Server protection

- Master NS
- Slave NS(s)
- Caching NS
- Resolver

Data protection

- Authentication and integrity of data (DNSKEY/RRSIG/NSEC)

- Server protection
  Authenticate communication between servers (TSIG/SIG0)

- Data protection
  Authenticate communication between servers (TSIG/SIG0)
What is DNSsec?

- Started 1995; RFC2535 published 1999
- Revised March 2005
  RFC4033, RFC4034, RFC4035
- Secure DNS addresses different issues:
  a. Authenticated zone transfer
     TSIG between authoritative name server (master/slave)
  b. Secure dynamic updates (RFC3007)
     TSIG/SIG0 between updater and master server
  c. Authenticated zone data
     Signed resource record sets
  d. Authenticated queries
- Implementation: bind-9.3.x, NSD 2.1.x
- Applications are more and more depending on DNS
  (Anti SPAM (MARID/MASS/DKIM), ENUM, SSH-Fingerprints, SRV-Records)
DNSsec specific Resource Records

- **TSIG (RFC2845)** Pseudo-RR
  Secret key transaction authentication for DNS (hashed MD5)
  Used by authenticated zone transfer and signed query Updates

- **TKEY (RFC2930)** Pseudo-RR
  Secret key establishment for DNS (Diffie-Hellman, Sig(0), GSSAPI)

- **SIG(0) (RFC2931)** Pseudo-RR
  DNS request and transaction signatures
  (Public-Key: RSA-MD5, RSA-SHA1, DSA)

- **RRSIG, DNSKEY, NSEC (RFC4034)**
  Old: SIG, KEY, NXT (RFC2535, RFC3845)
  Signed resource records (Public-Key)

- **DS (RFC3658)**
  Delegation signer resource record
DNSsec Overview

- Authenticated data origin
- Signing of the zone data
  No encryption; authenticity and integrity only
- Based on asymmetric keys
  - The private part is used to create the signature
  - The public part of the key is used for verification
- Build a chain of trust, up to the root
  The parent must be secure
- Provide the trust-anchor to the verifying resolver
  Zone status could be:
  - Verifiable Secure
  - Verifiable Insecure
  - Bad
DNSsec


RRSIG – Resource Record Signature

• Signed Resource Record Sets
  Every resource record is signed and the result is published via RRSIG Record

$ORIGIN example.net.
host 7200 IN A 1.2.3.4
7200 IN A 2.4.5.6
7200 IN RRSIG A 1 3 7200 (20050918041800 20050819041800 ; Sig. Lifetime
18140 example.net. ; Keytag+Name
AK9adL3Ov7VkJYoa/5CHUO...== ) ; Signature

host 7200 IN AAAA 2001:0db8:900:2af::2
7200 IN RRSIG AAAA 1 3 7200 (20050918041800 20050819041800
18140 example.net. Zq0+A2...==)

• Same is possible for reverse zones (in-addr.arpa, ip6.arpa) and e.g. e164.arpa.
• Signature: Hash over the RR Data, encrypted with private part of the asymmetric zone signing key
• To create a signature obviously we need a key
DNSKEY – Keys for zone signing

• The public part of the zone signing key is part of the zone

```
$ORIGIN example.net.
example.net. 7200 IN DNSKEY 256 3 3 (AQOfy1zMaX1b2qC financesma8GxTSr+ToCUB6Z6
0m0zdE4sly0zRrdmgktamXxX00Xx9FM1Dw37WI9npZ6R); key id = 42398
```

```
7200 IN DNSKEY 256 3 3 (AQO3OsR3JpgGm1EfwMDVmGLYZYvUvSB0ewBegU9EdKI3
Bwlf233G9fwO9nK8fErYhkabUWEo31XOAXR2BmqQRTw/); key id = 18140
```

```
7200 IN DNSKEY 257 3 5 (AQP1PstpDYkKzruSFKBIQmrKQbioPO9PE7GzlaZtk
/FKDqyRVgXaUSK13egIQgQ5LH8pJYZZqu1+YxXDhbVIJ
...
FdjdSkBnTsyS/tDWiD98Qho8+0gwbpRvRRu5sTg1GfIc
bBxrL2EvY8uP8ci0mOXbDGSNwY3PmHi+b9Vf); key id = 65100
```

• Oops, so many keys ...

• Two type of signing keys
  — Zone signing keys (ZSK) are used to sign the zone data
  — Key signing keys (KSK) sign the ZSK only

• More than one KSK/ZSK is used during key replacement
DS – Delegation Signer Record (Chain of Trust)

• Build a chain of trust: Secure Delegation

$ORIGIN net.
example 14400 IN NS ns1.example.net.
14400 IN NS ns2.example.net.
14400 IN DS 65100 5 (1 5AB7376A226EBEA87BFE490A0E55E02FA5FE9147)
14400 IN RRSIG DS 1 3 14400 (20050918041800 20050819041800 28040 net.
O/d4As9zzkN+fxjshohV1OY/aX38UvDzWA74
gleLD+uLuWpflV6D3XwWwvTYnqMHM5kuLnbM
EE1KtDml+0tQhA== )

• The delegation signer record (DS) is a pointer to the KSK of the delegated zone

• The DS is signed by the parent!

• The NS is not signed by the parent!
  The NS records are originated in the delegated zone.

• The DS is the only record which is solely allowed in the parent zone
Chain of Trust / Secure Entry Points

- Chain of Trust build with **DS Records**
- The resolver needs some „**Secure Entry Points**“ (aka: trust anchor)
  Ideally: Only one trust anchor necessary
Trusted Keys – Trust Anchor

- The resolver requires a trust anchor to verify the chain of trust
  Also known as secure entry point (SEP)

- Ideally this is the KSK of the root zone, but many SEPs are allowed

- Example: BIND as verifying resolver

```plaintext
options {
    recursion yes;
    dnssec-enable yes;
    edns-udp-size 4096; # this is the default!
};
trusted-keys {
    "example.net." 257 3 5 "
    AQP1PstpDYkKzruSFKBIQmrKQbiaoPI09PE7GVz1aZtk/FKDqyRVxGxU
    SKl3egIQqQ5LH8pJYZZqu1+YxXDhbVIJPXK/3E2uxaZ8yWn+BIYm3DbY
    lftBvRU3pzDEpOjfU2RrFR7H38hj+jQOEYLnZxRrmfS7PlSXDEYHdN3
    c2u0dXZmcVIGFLG0XAirr/ZJ0Mb2LAqvMRvhf9KSp5bTM/dNm0l6/WHE
    TPwQ/gjHj4fBoL2yjF/3IcAaQjd4LNPjzWJcDCd7FdjdSkBnTsyS/tDW
    iD98Qho8+0gwbpxvRRu5sTgLGFicbBxrL2EvY8uP8ciOxODGSNwY3P
    mHi+b9Vf" ; # key id = 65100
    "213.in-addr.arpa." 257 3 5 ".....";
};
```
$ dig +multiline +dnssec host.example.net
; <<>> DiG 9.3.1 <<>> +multiline +dnssec host.example.net
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1730
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 3, AUTHORITY: 5, ADDITIONAL: 11
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;max.hznet.de. IN A

;; ANSWER SECTION:
host.example.net. 14400 IN A 1.2.3.4
host.example.net. 14400 IN A 2.3.4.5
host.example.net. 14400 IN RRSIG A 1 3 14400 20050908041800 (20050809041800 18140 example.net. WeUoDexKRUCj3rqGICyi2X4U+w+/q3RGYUg1HtLLatyc pei813WTKqi2Jd1/v14KSfZLNYfWxr5DWu7jmgI7uQ== )

;; Query time: 103 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Aug 9 20:29:18 2005
;; MSG SIZE rcvd: 770
DNSsec Practice (Bind Tools)

- Create the key material
  - Key signing key (KSK)
    
    $\textit{dnssec-keygen -f KSK -n ZONE -a DSA -b 1024 example.net}$
  
  - Zone signing key (ZSK)
    
    $\textit{dnssec-keygen -n ZONE -a RSASHA1 -b 512 example.net}$

- Store the public part of the key in the zone file (ok, do this only once)
  
    $\textit{cat Kexample.net+*.key >> zone.db}$

- Increment SOA serial number (How?)

- Sign the zone file
  
    $\textit{dnssec-signzone -g -o example.net zone.db}$
    $\textit{zone.db.signed}$
DNSsec Practice (2)

• Configure named
  
  options {
    dnssec-enable yes:
  };

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

• Reload the zone
  
  $ rndc reload example.net

• Re-sign the zone before the signature times out
  But: Don’t forget to increment the serial number

• Start a key rollover if the lifetime of the key is over
  There are two different ways to do this
Key Rollover

- DNSSEC Operational Practices define two algorithms for key rollover

- ZSK Rollover (pre-publish key)
  1. Generate second ZSK
  2. Publish both (public) keys, but use only the old one for signing
  3. Wait at least propagation time + TTL of the key set
  4. Use new key for zone signing; leave old one published
  5. Wait at least propagation time + maximum TTL of the old zone
  6. Remove old key

- KSK Rollover (double signature)
  1. Generate new KSK
  2. Use both keys for key signing
  3. Send new DS-set to the parent
  4. Wait until the DS is propagated + maximum TTL of the old zone
  5. Remove the old key
DNSsec Tools

• KROd – Key Rollover Daemon (www.idsa.prd.fr/index.php?page=kro&lang=en)
  — Full automatic ZSK rollover
  — Full automatic KSK rollover
  incl. KSK key exchange with the parent domain

• DNSSEC Key Maintenance Tools (www.ripe.net/disi/code.html)
  — Secure private key storage
  — Semi-automatic [KZ]SK rollover (pre-publish & double signature)

• DNSsec Tools (www.dnssec-tools.org)
  — Zone signing and key management tool

• Zone Key Tool (www.hznet.de/zkt/)
  — Automatic ZSK rollover
  — Full automatic re-signing of the zone (incl. SOA incrementation)
  — Parses secure zones out of named.conf
Zone Key Tool (ZKT)

- Provides Tools for key management and zone signing
  
  ```
  $ dnssec-zkt
  $ dnssec-signer -N /etc/named.conf
  ```

- Simple configuration file (extract of `dnssec.conf`)
  
  ```
  # zone specific timing values
  ResignInterval: 3d  # (259200 seconds)
  Sigvalidity: 30d  # (2592000 seconds)
  Max_TTL: 6h  # (21600 seconds)
  Propagation: 5m  # (300 seconds)

  # signing key parameters
  KSK_lifetime: 0
  KSK_algo: DSA  # (Algorithm ID 3)
  KSK_bits: 1024
  ZSK_lifetime: 10d  # (864000 seconds)
  ZSK_algo: RSASHA1  # (Algorithm ID 5)
  ZSK_bits: 512
  ```

- Full automatic ZSK rollover (pre-publish key algorithm)
- Automatic serial number incrementation
  Supports sequential serial number and YYYYmmDDxx Format
ZKT – Configuration

- Create a directory for each secure zone (dirname = domainname)
  
  ```
  $ mkdir example.net.
  $ cd example.net.
  ```

- Create the zone file (default name: 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 spaces!!
  ; 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
  ```
• **Create a (just empty) zone.db.signed file**

```bash
$ touch zone.db.signed
$ ls -l
-rw-r----- 1 dnsop dnsop 916 2005-08-14 13:54 zone.db
-rw-r--r-- 1 dnsop dnsop  0 2005-08-14 13:55 zone.db.signed
```

• **Sign the zone**

```bash
$ 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 2005-08-14 13:55 Kexample.net.+003+18710.key
-rw------- 1 dnsop dnsop 688 2005-08-14 13:55 Kexample.net.+003+18710.private
-rw-r--r-- 1 dnsop dnsop 121 2005-08-14 13:55 Kexample.net.+005+57705.key
-rw------- 1 dnsop dnsop 545 2005-08-14 13:55 Kexample.net.+005+57705.private
-rw-r--r-- 1 dnsop dnsop 1136 2005-08-14 13:55 dnskey.db
-rw-r----- 1 dnsop dnsop   7 2005-08-14 13:55 dsset-example.net.
-rw-r--r-- 1 dnsop dnsop  702 2005-08-14 13:55 keyset-example.net.
-rw-r----- 1 dnsop dnsop  916 2005-08-14 13:55 zone.db
-rw-r--r-- 1 dnsop dnsop 4080 2005-08-14 13:55 zone.db.signed
```
ZKT – Configuration(3)

- **Show current key status**
  
  ```
  $ dnssec-zkt -a .
  
  Keyname  Tag Typ Sta   Algorit  Generation Time Age
  example.net.  18710 KSK act  DSA  Aug 14 2005 13:55:24 13m42s
  example.net.  57705 ZSK act RSASHA1 Aug 14 2005 13:55:24 13m42s
  ```

- **Change the zonefile in named.conf**
  
  ```
  zone "example.net." in {
    type master;
    file "example.net./zone.db.signed";
  }
  ```

- **Force re-signing and reload the zone**
  
  ```
  $ dnssec-signer -r -f -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."
  ```

- **Check messages in /var/log/named**
  
  ```
  14-Aug-2005 14:34:43.198 general: info: zone example.net/IN: loaded serial 65 (signed)
  ```
ZKT – Configuration(4)

- **Periodic re-sign your zone**
  Call `dnssec-signer` at least once a day

- **`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
  ```

- **The `dnssec-cron` script looks simple**

  ```
  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
  ```

- **Create the trusted-keys Section for your resolver configuration**

  ```
  $ dnssec-zkt -T -l example.net.
  trusted-keys {
    "example.net."   257 3 3 "CJEUcyN1ES5bAnBi4O+m7nLhbmtfxVtF31O4agNVe+6Hu8kZ8EKzm+/U
      +qh2NXv6+UgowadnP1fHHwLzpFP4aZXfXa2qog1P5dp7P0UquW6zn25
      Wdlf/F/21Jh2LF4bU616EyOeRichLv1BXn15nkkLr4usbPitr68DrVas
      o6bci4LJ1PJBkHVS/3MtBo01SY3XvoiBJtgp" ; # key id = 18710
  }
  ```
References


Nominum
BIND v9 Administrator Reference Manual

Olaf Kolkman, Ripe-NCC DISI „DNSSEC Howto Version 1.3“

RFCs 1034, 1035, 2535, 2848, 2930, 2931, 3007, 3655, 3658, 3757, 3833, 3845
  4033 (DNS Security Introduction and Requirements)
  4034 (Resource Records for the DNS Security Extentions)
  4035 (Protocol Modifications for the DNS Security Extensions)

Drafts DNSSEC Operational Practices
draft-ietf-dnsop-dnssec-operational-practices-04.txt

Links http://www.dnssec.net
http://www.hznet.de/dns/dnssec-denic040929.pdf
Questions ?
Questions ?

http://www.hznet.de/dns/dnssec-decix050916.pdf
Questions?

http://www.hznet.de/dns/dnssec-decix050916.pdf

Thank you very much for your attention!
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agenda</td>
<td>1</td>
</tr>
<tr>
<td>Why DNSsec ?</td>
<td>2</td>
</tr>
<tr>
<td>What is DNSsec ?</td>
<td>3</td>
</tr>
<tr>
<td>DNSsec specific Resource Records</td>
<td>4</td>
</tr>
<tr>
<td>DNSsec Overview</td>
<td>5</td>
</tr>
<tr>
<td>RRSIG – Resource Record Signature</td>
<td>6</td>
</tr>
<tr>
<td>DNSKEY – Keys for zone signing</td>
<td>7</td>
</tr>
<tr>
<td>DS – Delegation Signer Record (Chain of Trust)</td>
<td>8</td>
</tr>
<tr>
<td>Chain of Trust / Secure Entry Points</td>
<td>9</td>
</tr>
<tr>
<td>Trusted Keys – Trust Anchor</td>
<td>10</td>
</tr>
<tr>
<td>Stub-Resolver (dig)</td>
<td>11</td>
</tr>
<tr>
<td>DNSsec Practice (Bind Tools)</td>
<td>12</td>
</tr>
<tr>
<td>DNSsec Practice (2)</td>
<td>13</td>
</tr>
<tr>
<td>Key Rollover</td>
<td>14</td>
</tr>
<tr>
<td>DNSsec Tools</td>
<td>15</td>
</tr>
<tr>
<td>Zone Key Tool (ZKT)</td>
<td>16</td>
</tr>
<tr>
<td>ZKT – Configuration</td>
<td>17</td>
</tr>
<tr>
<td>ZKT – Configuration(2)</td>
<td>18</td>
</tr>
<tr>
<td>ZKT – Configuration(3)</td>
<td>19</td>
</tr>
<tr>
<td>ZKT – Configuration(4)</td>
<td>20</td>
</tr>
<tr>
<td>References</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>23</td>
</tr>
</tbody>
</table>