

# DNSSEC

## Zone Management

### with ZKT

DENIC DNSSEC Testbed Workshop

Frankfurt/Main Germany  
26. Jan 2010

*Holger.Zuleger@hznet.de*

# Agenda

- ZKT Overview
  - Intro
  - Key features
- DNSSEC Basics
  - Key generation
  - dnssec-zkt
  - Zone signing
  - dnssec-signer
- ZKT system setup
  - Logging
  - Running on master
  - Running on stealth master
  - ZKT without BIND name server
- DNSSEC in 6 Minutes

# Zone Key Tool

- One of the first open source tools for dnssec management  
zkt-0.5 released Apr 1 2005; current release is zkt-0.99c
- Wrapper around BIND dnssec commands
- Contributed with BIND since 9.6.0a1
- Download via sourceforge or the project website  
<http://sourceforge.net/projects/zkt>, <http://www.hznet.de/dns/zkt/>
- FreeBSD and OpenBSD ports available  
Maintained by Frank Behrens and Jakob Schlyter
- zkt-users mailing list (low volume)  
<https://lists.sourceforge.net/lists/listinfo/zkt-users>
- Supports all newer BIND versions (9.3 up to 9.6)  
BIND 9.7 is supported but currently w/o the fancy new dnssec features

# ZKT Key Features

- DNSKEY state list command
- Automated key generation and rollover
  - Pre-Publish (RFC4641 used for ZSK rollover)
  - RFC5011 (KSK rollover)
  - Double Signature (RFC4641 for KSK rollover)
- Automatic serial number change
  - UNIX time (seconds since the epoch)
  - plain integer
  - date format (yyyymmddnn)
- Signs „full zone“ at once  
Per RR signing for dynamic zones only
- Dynamic zone signing support (still experimental)  
Wait for BIND 9.7 for better dynamic zone support
- File and syslog based logging

- ZKT Overview
  - Intro
  - Key features
- DNSSEC Basics
  - Key generation
  - dnssec-zkt
  - Zone signing
  - dnssec-signer
- ZKT system setup
  - Logging
  - Running on master
  - Running on stealth master
  - ZKT without BIND name server
- DNSSEC in 6 Minutes

# DNSSEC Key Generation

- BIND uses two commands for DNSSEC maintenance

- a. `dnssec-keygen` for key generation
- b. `dnssec-signzone` for zone signing

- `dnssec-keygen` requires some options to generate a DNSSEC key

```
$ dnssec-keygen -a RSASHA1 -b 1300 -r /dev/urandom -n ZONE -f KSK example.net.
```

- ZKT uses a wrapper command plus a config file to simplify this:

```
$ sed -n "/signing/,/^$/p" dnssec.conf
# signing key parameters
Key_algo:      RSASHA1    # (Algorithm ID 5)
KSK_lifetime:  1y         # (31536000 seconds)
KSK_bits:      1300
KSK_randfile:  "/dev/urandom"
ZSK_lifetime:  12w        # (7257600 seconds)
ZSK_bits:      512
ZSK_randfile:  "/dev/urandom"
SaltBits:      24
```

```
$ dnssec-zkt --ksk --create example.net. ; generate a ksk
$ dnssec-zkt -z -C example.net.         ; generate a zsk
```

# DNSSEC keyfiles

- A DNSSEC key is represented by two files (**public** and **private** part)

```
$ ls -l K*
-rw-r--r-- 1 dns dnsadm 313 2009-12-07 00:57 Kexample.net.+005+27450.key
-rw----- 1 dns dnsadm 1157 2009-12-06 17:43 Kexample.net.+005+27450.private
-rw-r--r-- 1 dns dnsadm 177 2009-11-15 16:40 Kexample.net.+005+54680.key
-rw----- 1 dns dnsadm 553 2009-11-15 18:40 Kexample.net.+005+54680.private
```

- Key infos are part of the filesystem (**zone**, **tag**, **algorithm**, **date**)  
But the type of key is coded in the flags field only

- `dnssec-zkt` is able to list DNSKEYs in a user friendly form

```
$ dnssec-zkt -a -t -f
Keyname          Tag Typ Sta Algorit          Age Lftm
example.net.    27450 KSK act RSASHA1      1w 2d23h34m55s<365d
example.net.    54680 ZSK act RSASHA1      4w 3d 7h52m17s!28d
```

- Some of the options are setable via the config file

```
$ sed -n '/zkt options/,/^$/p' /var/named/dnssec.conf | grep ":"
Zonedir:          "/var/named"
Recursive:        True
PrintTime:        True
PrintAge:         False
LeftJustify:      False
```

## dnssec-zkt

- List all DNSKEYs (a bit like `ls` for files)  
Sorted by domain name, key type (KSK, ZSK) and date
- List keys in subdirectories recursively (Option `-r`)
- A directory or a key file could be specified as argument  
Default directory is settable via `zonedir` in `dnssec.conf`.
- Option `-p` prints the path name where the key file was found

```
$ dnssec-zkt -r -p -l example.net. .
Keyname                               Tag Typ Sta Algorit Generation Time
./views/intern/example.net./
    example.net. 00126 KSK act RSASHA1 Nov 20 2009 12:44:27
./views/extern/example.net./
    example.net. 23553 KSK act RSASHA1 Nov 20 2009 12:49:04
./views/intern/example.net./
    example.net. 05972 ZSK act RSASHA1 Nov 20 2009 12:44:27
./views/extern/example.net./
    example.net. 36122 ZSK act RSASHA1 Nov 20 2009 12:49:05
    example.net. 35744 ZSK pre RSASHA1 Dec 17 2009 23:45:27
```



## dnssec-zkt (example output)

- Recursive key listing (sorted by domain name, key type and age)

```
$ dnssec-zkt -r -a examples
Keyname          Tag Typ Sta Algorit Generation Time          Age
sub.example.de.  27321 KSK act RSASHA1 Apr 15 2009 18:40:55 5w 1d 6h36m23s
sub.example.de.  23742 ZSK pre RSASHA1 May 11 2009 23:32:01 1w 3d 1h45m17s
sub.example.de.  29194 ZSK act RSASHA1 May 09 2009 14:05:34 1w 3d 1h45m17s
example.de.     58635 KSK rev RSASHA1 Apr 23 2009 18:10:22 4w   9h 6m56s
example.de.     27450 KSK act RSASHA1 May 06 2009 17:43:29 2w 1d 19m56s
example.de.     17439 KSK sta RSASHA1 May 07 2009 00:57:22 2w 1d 19m56s
example.de.     54680 ZSK act RSASHA1 Apr 15 2009 18:40:55 5w 1d 8h37m18s
dyn.example.net. 09399 KSK act RSASHA1 May 16 2009 12:39:19 5d12h37m59s
dyn.example.net. 46577 ZSK act RSASHA1 May 16 2009 12:39:19 5d12h37m59s
sub.example.net. 54876 KSK act RSASHA1 Oct 01 2008 08:24:24 33w 2d16h52m54s
sub.example.net. 01646 ZSK act RSASHA1 May 09 2009 13:59:11 1d13h47m16s
sub.example.net. 26431 ZSK dep RSASHA1 May 06 2009 15:25:28 1d13h47m16s
example.net.    41151 KSK act RSASHA1 Apr 20 2009 22:54:22 4w 3d 2h22m56s
example.net.    01764 KSK sta RSASHA1 May 06 2009 23:26:34 2w 1d 1h37m 3s
example.net.    05972 ZSK act RSASHA1 Nov 20 2007 12:44:27 26w 1d11h32m51s
```

- The state of a key is represented by the private key file name
  - e.g. state is pre-published (or standby) if private key file ends in `.published`
  - **First stage** of ZSK rollover: `sub.example.de`
  - **Last stage** of ZSK rollover: `sub.example.net`
  - **rfc5011 KSK** rollover in place: `example.de`

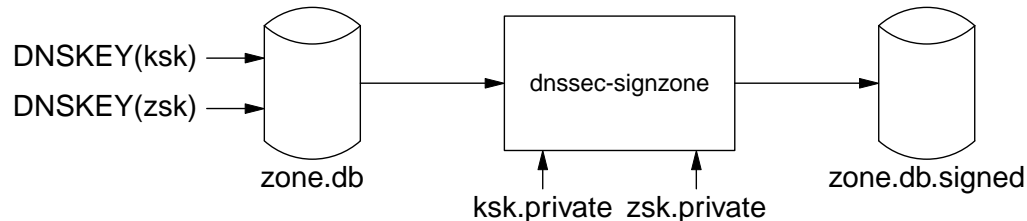
## dnssec-zkt (Build in defaults)

- Print out the build in config options

```
$ dnssec-zkt -c /dev/null -Z
#      @(#) dnssec.conf vT0.99d (c) Feb 2005 - Aug 2009 Holger Zuleger hznet.de
#   dnssec-zkt options
Zonedir:      "."
Recursive:    False
PrintTime:    True
PrintAge:     False
LeftJustify:  False
#   zone specific values
ResignInterval: 1w      # (604800 seconds)
Sigvalidity:  10d     # (864000 seconds)
Max_TTL:      8h      # (28800 seconds)
Propagation:  5m      # (300 seconds)
KEY_TTL:      4h      # (14400 seconds)
Serialformat: incremental
#   signing key parameters
Key_algo:     RSASHA1 # (Algorithm ID 5)
KSK_lifetime: 1y      # (31536000 seconds)
KSK_bits:     1300
KSK_randfile: "/dev/urandom"
ZSK_lifetime: 12w     # (7257600 seconds)
ZSK_bits:     512
ZSK_randfile: "/dev/urandom"
SaltBits:     24
#   dnssec-signer options
LogFile:      ""
LogLevel:     ERROR
SyslogFacility: NONE
SyslogLevel:  NOTICE
VerboseLog:   0
Keyfile:      "dnskey.db"
Zonefile:     "zone.db"
DLV_Domain:  ""
Sig_Pseudorand: False
Sig_GenerateDS: True
Sig_DnsKeyKSK: False
Sig_Parameter: ""
```

## Zone signing with BIND

- BIND provides the `dnssec-signzone` command for zone signing  
The command adds RRSIG and NSEC/NSEC3 records to a zone



- Please add DNSKEY RRs to the zone file before signing  
At best via `$INCLUDE` directive
- Corresponding `.private` files in the curr dir will be used for signing
- Lifetime of RRSIG record is configurable (default is 30 days)  

```
$ dnssec-signzone -r /dev/urandom -g -e +172800 -o example.net zone.db
```
- Please increment the SOA serial number before signing  
Since bind 9.4 serial number is settable with „-N unixtime“ to the current time
- You have to resigning the zone before the signature expire

## Zone signing with ZKT

- A wrapper command (`dnssec-signer`) is used for zone signing
  - The wrapper increments the serial number by itself
    - Supports plain integer, YYYYMMDDnn and unixtime (BIND9.4) format
  - Adds all necessary DNSKEY RR into a „database“ (`dnskey.db`)
    - Please include this file into the zonefile `$INCLUDE dnskey.db` file
  - Signing parameters are specified via `dnssec.conf` file
  - Starts the signing process only if needed (a bit like `make`)
    - Update on zone file, refresh of RRSIG, new keys added, etc.
  - Triggers a key rollover if necessary
- Example Zone file

```
$TTL      7200
;         The serial number is left justified in a field of at least 10 chars!!
@         IN SOA  ns1.example.net. hostmaster.example.net. (
                                244           ; Serial
                                43200        ; Refresh
                                1800        ; Retry
                                2W          ; Expire
                                7200 )     ; Minimum

$INCLUDE dnskey.db
```

## dnssec-signer options

- Be silent or very verbose (`-v -v`)
- Option `-f` forces a re-signing of the zone
- Four „mode of operation“
  - a. Signing of a single zone (in the current directory)

```
$ dnssec-signer -v -v -o example.net.
```
  - b. Signing of all zones below „Zonedir“

```
$ dnssec-signer -v -v
```
  - c. Sign all master zones (`.signed`) in `named.conf`

```
$ dnssec-signer -v -v -N /var/named/named.conf
```
  - d. Signing of all zones below a given directory

```
$ dnssec-signer -v -v -D /var/named/zones/de.
```
- Use option `-r` to trigger a reload of the zone (via `rndc`)  
The reload will be triggered only if it's necessary (new `.signed` file)

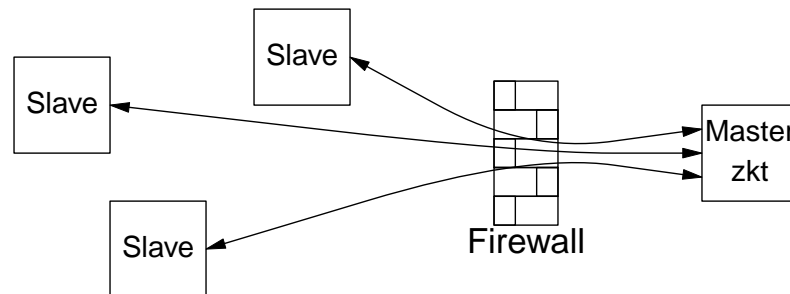
- ZKT Overview
  - Intro
  - Key features
- DNSSEC Basics
  - Key generation
  - dnssec-zkt
  - Zone signing
  - dnssec-signer
- ZKT system setup
  - Logging
  - Running on master
  - Running on stealth master
  - ZKT without BIND name server
- DNSSEC in 6 Minutes

## ZKT logging

- Logging is independent of (verbose) output to stdout  
But could be logged as well (Verboselog: 0|1|2)
- Several log level supported  
debug, info, notice, warning, error, fatal; „none“ turns logging of
- Logging to a log *file* or *directory* (LogFile, LogLevel)
  - A file will be overwritten by each dnssec-signer run
  - If a directory is specified, different files for logging will be used
  - File name looks like `zkt-YYYY-mm-ddThhmmssZ.log` (UTC)
- Syslog logging with configurable facility and loglevel
- Exit code of dnssec-signer reflects number of errors:
  - 0: no error
  - 1-64: The number of errors occurred (or more than 63)
  - 127: Fatal error

## ZKT / Bind Setup

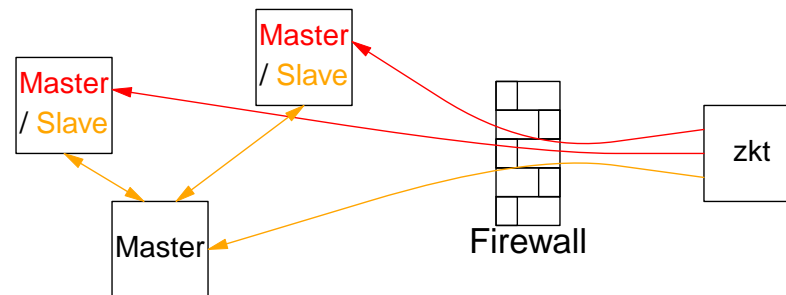
- The most simple setup:
  - ZKT is running on the master name server
  - Every zone has it's own directory with all the keys in it
  - Zone transfer to slave name server via [A|I]XFR
  - This setup is also good for dynamic zones
- If this is too risky use a stealth master (hidden primary) setup
  - ZKT is running on the stealth master
  - Master is behind a firewall
  - Zone transfer to visible slave name server via [A|I]XFR
  - More secure but also a bit more complex





## ZKT running w/o BIND

- Run ZKT on your provisioning system without a name server
- Provisioning system is behind a firewall
- Send signed zone files to master name server via (e.g.) `scp`
- Reload zone via `ssh` or `rndc`
- **Single-** or **multi** master setup possible



- Configure ZKT to use a distribute command

```
Distribute_Cmd: /var/named/distribute.sh
```

Example script is part of zkt

- ZKT Overview
  - Intro
  - Key features
- DNSSEC Basics
  - Key generation
  - dnssec-zkt
  - Zone signing
  - dnssec-signer
- ZKT system setup
  - Logging
  - Running on master
  - Running on stealth master
  - ZKT without BIND name server
- **DNSSEC in 6 Minutes**

# DNSSEC in 6 Minutes

(by Alan Clegg from ISC: [http://www.isc.org/files/DNSSEC\\_in\\_6\\_minutes.pdf](http://www.isc.org/files/DNSSEC_in_6_minutes.pdf))

- Create a sidewise `dnssec.conf` file (e.g. for unix timestamp support)

```
$ cd /var/named
$ dnssec-zkt -O "Serialformat: unixtime; Zonedir: /var/named/zones" -Z > dnssec.conf
```

- Create a new directory and copy your existing zone file into it

```
$ mkdir -p zones/example.de; cd zones/example.de
$ cp /var/named/db.example.de zone.db
```

- Include the dnskey database into the zone file

```
$ echo "\$INCLUDE dnskey.db" >> zone.db
```

- Create an (even empty) signed zone file

```
$ cat /dev/null > zone.db.signed
```

- Change the name of the zone file in `named.conf`

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

## DNSSEC in 6 Minutes (2)

- Run the ZKT signer command and reload the zone

```
$ dnssec-signer -r -v -o example.de.  
parsing zone "example.de." in dir "."  
  Check RFC5011 status  
  Check KSK status  
  No active KSK found: generate new one  
  Check ZSK status  
  No active ZSK found: generate new one  
  Re-signing necessary: Modified zone key set  
  Writing key file "./dnskey.db"  
  Signing zone "example.de."  
  Signing completed after 0s.  
  Reload zone "example.de."
```

- Run the signer command in regular intervals  
Use cron for this, but turn off verbose output

```
$ dnssec-signer -r -v -N /var/named/named.conf  
parsing zone "example.de." in dir "/var/named/zones/example.de"  
  Check RFC5011 status  
  Check KSK status  
  Check ZSK status  
  Re-signing not necessary!
```

- Done!

# Fragen ?

H Z N E T

DNSsec, VoIPsec, IPsec, XMPPsec, SMTPsec, WLANsec ...

... DKIM, Kerberos, IMAP, LDAP, ENUM, SIP, ...

... NTP, DNS, DHCP, IPv6, Routing, Switching

Holger.Zuleger@hznet.de

# Backup Slides

## Double Signature (KSK) Rollover

- If lifetime of KSK is over
  1. Generate a new KSK; Use both ksk for key signing  
Wait until new key is known by resolver (propagation time + old key TTL)
  2. Send new DSset (or keyset) to the parent  
Wait until the DS is propagated + TTL of the old DS-RR
  3. Remove the old key
- Step two is under discussion  
How to send the DS or DNSKEY to the parent?
- Automatic ksk rollover if parent is under control of zkt on the same host  
Use a hierachical directory structure with child dir below the parent
- Otherwise a warning message is written into logfile if ksk is expired
- Use manual KSK rollover feature of dnssec-zkt then
  - \$ dnssec-zkt --ksk-roll-phase1 example.net.      \$ dnssec-zkt --ksk-newkey example.net.
  - \$ dnssec-zkt --ksk-roll-phase2 example.net.      \$ dnssec-zkt --ksk-publish example.net.
  - \$ dnssec-zkt --ksk-roll-phase3 example.net.      \$ dnssec-zkt --ksk-delkey example.net.

## RFC5011 KSK rollover

- Create a „standby“ key manually  
A standby key is a pre published KSK not used for signing  
\$ dnssec-zkt --ksk --create test.example.net.
- If the lifetime of the active KSK is over
  - a. A new standby key will be created
  - b. The old standby key will be activated
  - c. The old active key will be revoked
  - d. After 30 days, revoked key will be removed from zone apex

- Revoking a key means to set **bit 8** in the flags field

```
example.de. IN DNSKEY 385 3 5 BQEAAAABDAEYYP2lsGo...= ;
                /      \
                110000001
```

- Pay attention: Changing the flag field results in a new key tag(id)

```
$ dnssec-zkt --nohead --list-dnskey --ksk -l example.de.
example.de. IN DNSKEY 385 3 5 (
    BQEAAAABDAEYYP2lsGob0e77EYYDqsr ... wQ==
) ; key id = 58763 (original key id = 58635)
```



## CONTENTS

.....	1
Agenda .....	2
Zone Key Tool .....	3
ZKT Key Features .....	4
.....	5
DNSSEC Key Generation .....	6
DNSSEC keyfiles .....	7
dnssec-zkt .....	8
dnssec-zkt (example output) .....	9
dnssec-zkt (Build in defaults) .....	10
Zone signing with BIND .....	11
Zone signing with ZKT .....	12
dnssec-signer options .....	13
.....	14
ZKT logging .....	15
ZKT / Bind Setup .....	16
ZKT running w/o BIND .....	17
.....	18
DNSSEC in 6 Minutes .....	19
DNSSEC in 6 Minutes (2) .....	20
.....	21
.....	22
Double Signature (KSK) Rollover .....	23
RFC5011 KSK rollover .....	24