Step-by-Step DNSSEC-Tools Operator Guidance Document

Using the DNSSEC-Tools v1.0 distribution

SPARTA, Inc.

Table of Contents

List of Tables

Chapter 1. Introduction

DNS Security (DNSSEC) helps protect against DNS-spoofing attacks by providing origin authentication and integrity protection of DNS information. Proper maintenance of a DNSSEC-enhanced DNS zone is essential to protecting the domain's zone data.

This Step-by-Step DNSSEC-Tools Operator Guidance Document is intended for operations using the DNSSEC-Tools v1.0 distribution. It will assist operators in gaining operational experience with DNSSEC. Some basic understanding of DNSSEC terms and concepts is required. It follows the format laid out by [dnssec-operators-guide].

This document is meant to be a learning aid and is not intended to define policy in any form. Any implicit recommendations for key sizes, signature validity periods, and command line parameters are for illustration purposes ONLY and MUST NOT be used in production environments unless due-diligence has been taken to ensure that these values are acceptable within such environments. See [dnssec-operational-practices] for suggestions on determining appropriate security characteristics.

This document was written as part of the DNSSEC-Tools project. The goal of this project is to create a set of documentation, tools, patches, applications, libraries, wrappers, extensions, and plug-ins that will help ease the deployment of DNSSEC-related technologies. For more information about this project and the tools that are being developed and provided, please see the DNSSEC-Tools project web page at: `http://www.dnssec-tools.org`.

Organization of this Document

This guide contains the following sections.

Section 1. Introduction to the Step-By-Step Guide.

Section 2. Describes the configuration required before the DNSSEC-Tools utilities may be used.

Section 3. Describes how to perform an initial signing of a zone.

Section 4. Provides the steps required to configure a name server to serve a signed zone.

Section 5. Gives information on checking for expiration of a zone's signatures.

Section 6. Describes how to re-sign a previously signed zone.

Section 7. Provides the commands required for a child zone to create a signed delegation.

Section 8. Gives the commands required for a parent zone to create a signed delegation.

Section 9. Describes the Pre-Publish Scheme, which is used in rollover operations of ZSK keys.

Section 10. Provides the Double-Signature Scheme, which is used in rollover operations of KSK keys.

Section 11. Gives the emergency rollover procedures to take in the event of a ZSK key compromise.

Section 12. Describes the emergency rollover procedures to take in the event of a Published ZSK key compromise.

Section 13. Provides the emergency rollover procedures to take in the event that both the Published and Current ZSK keys are compromised.

Section 14. Gives the emergency rollover procedures to take if the KSK key is compromised.

Section 15. Describes the actions a parent zone must take when a child zone's KSK key is compromised.

Section 16. Provides a migration path for moving to using the DNSSEC-Tools toolset.

Section 17. Gives information on configuring a secure resolver.

Key Concepts

A number of concepts must be known in order to understand this document.

Zones and Authentication Keys

Zones and Authentication Keys are essential for understanding this document, but they are also beyond its scope.

Zone Rollover

As zone signatures expire, the zone must be re-signed with new keys. The process of generating new keys and re-signing the zone is called *zone rollover*. There are several rollover schemes (e.g., Double-Signature Scheme and Pre-Publish Scheme) that are used for various purposes. These schemes are described in Chapter 10, *KSK Rollover (Double-Signature Scheme)* and Chapter 9, *Current ZSK Rollover (Pre-Publish Scheme)*.

Key-Tag Tables

The Key-Tag Table is a record of zones, the zone's keys, attributes of the keys, and expiration dates. This may be kept in any usable form -- computer file, notebook, etc.

Keyrec Files

Keyrec files function as Key-Tag Tables for DNSSEC-Tools utilities. They can be hand-edited, but the DNSSEC-Tools update them automatically.

Rollrec Files

Rollrec files contain information needed by the DNSSEC-Tools key rollovers. They can be hand-edited, but the DNSSEC-Tools update them automatically.

Conventions Used in this Document

One of the goals of this document is to self-contain DNS Security operations within sections and prevent constant cross-referencing between sections. Consequently, certain parts of the text are repeated throughout the document.

Text marked in bold represents text or commands entered by users within a given procedural step.

Underlined text, which can also be bold, is a place-holder for actual run-time values. These values are either automatically generated or are values that are known to the user from some other step.

Additionally, the following typographical conventions are used in this document.

Table 1.1. Conventions

command	Command names

`filename`	File and path names
`URL`	Web URLs
execution	Simple command executions

Longer sets of command sequences are given in this format:

```
# cd /tmp [ENTER]
# ls [ENTER]
# rm -fr * [ENTER]
#
```

In most cases, output will not be displayed for given command sequences.

Acknowledgments

This document builds upon the procedures laid out in [dnssec-operators-guide].

Comments

Please send any comments and corrections to sparta-dnssec@tislabs.com.

Chapter 2. Configure DNSSEC-Tools

The following sections must be read before proceeding with the rest of this guide.

The steps in the section called “Check for Randomness” and the section called “Create the DNSSEC-Tools Configuration File” *MUST* be performed prior to following any other steps.

Check for Randomness

Key generation and zone signing require random data to create strong cryptographic material. The **zonesigner** command defaults to using random data from `/dev/random`. Use this test to verify that `/dev/random` will provide data when requested:

 # dd if=/dev/random bs=2 count=10 | od -x `[ENTER]`

The above command checks if `/dev/random` is able to provide data when queried; it does not check to see that the data provided is truly random.

If this command provides data immediately, `/dev/random` will provide the data you need. If it hangs, then **zonesigner** won't be able to retrieve data, random or otherwise, from `/dev/random`.

If this check for randomness fails, pseudorandom numbers can be used instead. However, using pseudorandom numbers negatively affects the quality of the cryptographic material to a significant degree. A more appropriate measure would be to run **zonesigner** on a different system that has `/dev/random` and the ability to generate good random data.

Create the DNSSEC-Tools Configuration File

The DNSSEC-Tools configuration file contains many settings for customizing the DNSSEC-Tools suite of programs. The setting include things such as default authentication algorithm, directory for archived authentication keys, paths to various helper programs, and lengths of authentication keys. Configuration entries are in a *keyword/value* format. The keyword is a character string and the value is data associated with that keyword. `/usr/local/etc/dnssec/dnssec-tools.conf` is the default location for the configuration file.

The **dtinitconf** command will create a new DNSSEC-Tools configuration file. Command options will allow for automatic customization of the file. It is a plain text file, so any normal text editor (e.g., **vi** or **emacs**) may be used to modify the configuration file.

Several example option settings are given below. The man page for `dnssec-tools.conf` should be consulted for a complete list of possible options. Each option has a recommended setting, but that setting should not be considered a universally correct setting.

Table 2.1. DNSSEC-Tools Configuration Options

Option	Description	Recommended Setting
algorithm	The cryptographic algorithm to use for the keys.	rsasha1
endtime	The lifetime of the signatures.	+2592000 (30 days)
ksklength	The length of the KSK key.	2048
zsklength	The length of the ZSK key.	1024

BIND Name Server Execution

This document assumes that the BIND name server is executing. The specific command arguments are site-specific, so the BIND Administrator's Guide should be consulted.

Protect Your Files!

All rollrec files, keyrec files, zone files, and authentication keys **MUST** be properly protected. If these files are not protected, then the security of the zone files may be compromised.

- The `.private` portions of key files must only be readable or writable by the root user.
- The DNSSEC-Tools files must only be writable by the root user.

Chapter 3. Initially Signing a Zone

A zone must be signed before any other DNSSEC-Tools-related actions may be taken with it. This section describes how to sign a zone for the first time.

If a zone has been signed, it must be resigned when *any* change is made to it. Follow Chapter 6, *Resigning a Zone* when resigning a zone.

Sign the Zone with zonesigner

 # zonesigner -genkeys -gends -zone zone-name zone-file output-file [ENTER]

Key generation and signing may take a few minutes to complete depending on the size of the zone file and size of the keys. This operation may appear to be unresponsive for a period of time, depending on the operating system's random number generator device. (See Chapter 2, *Configure DNSSEC-Tools* for more information on random number generators and DNSSEC-Tools.)

The output is a set of files outlined below.

Table 3.1. zonesigner Output Files

File	Description
`output-file.signed`	The signed zone file. The .signed is added by **zonesigner**.
`keyset-zone-name`	The keyset for the zone. This is stored in the directory specified by the configuration file and may have to be sent to the parent zone - see Chapter 7, *Creating a Signed Delegation - Child Zone Activity*.
`dsset-zone-name`	The dsset for the zone. This is stored in the directory specified by the configuration file and may have to be sent to the parent zone - see Chapter 7, *Creating a Signed Delegation - Child Zone Activity*.
zone-name.krf	The keyrec file. This is used by **zonesigner** to maintain information about the keys used for the zone.
`Kzone-name.+algid+keytag.private`	The private key file. This is stored in the directory specified by the configuration file. The `keytag` is a unique identifier for this key. The *algid* is the numeric authentication algorithm identifier.
`Kzone-name.+algid+keytag.key`	The public key file. This is stored in the directory specified by the configuration file. The `keytag` is a unique identifier for this key. The *algid* is the numeric authentication algorithm identifier.

See the **zonesigner** man page for more information about the **zonesigner** command and its options.

Chapter 4. Configuring and Serving a Signed Zone

Several configuration files must be modified in order to serve a signed zone. Follow the steps below to configure your name server and have it start serving your signed zone.

`named.conf` is the name of the configuration file used in these examples. The configuration file may vary according to the needs of the administrator.

Add the Signed Zone to the Name Server Configuration File

The name of the signed zone file must be included in the name server's configuration file. If you are signing an existing zone, the current zone file in the configuration file must be replaced with the signed zone file. If you are signing a new zone, the new signed zone file must be added.

For the zone whose name is `zone-name`, do the following:

 # vi named.conf [ENTER]

```
... zone "zone-name." { type master; file "zone-file.signed"; }; ...
```

Enable DNSSEC

Add the `dnssec-enable yes;` option to the `named.conf` file.

 # vi named.conf[ENTER]

```
... options { ... dnssec-enable yes; }; ...
```

Check the Name Server Configuration File for Errors

You must ensure that the configuration file modifications were performed correctly. The **named-checkconf** command will perform this verification. No output indicates that all is well with the zone.

 # named-checkconf named.conf [ENTER]

Reload the Zone

The **rndc** command will reload the name server configuration files and zone contents. The name server process is assumed to be already running.

 # rndc reload zone-name [ENTER]

Check that the Zone Loaded Properly

Confirm that the SOA serial number of the zone corresponds to the most recent value.

 # dig @server-IP-address SOA zone-name [ENTER]

```
; <<>> DiG 9.3.0 <<>> ... ... ;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1,
ADDITIONAL: 0 ... ;; ANSWER SECTION zone-name 3600 IN SOA servername contact (
```

```
2005092101 ; This should be the most recent value. ; This value will most likely be
different in your zone file. ... ) ...
```

Chapter 5. Checking Signature Expiration

It is important to regularly check your zone for signatures that are nearing expiration. If the signatures are close to expiring, or already have expired, see Chapter 6, *Resigning a Zone* for how to resign the zone.

Check the Zone for Expiring Signatures

 # expchk -all -warn 10 keyrec-file `[ENTER]`

This checks the keyrec file to see if the zone has signatures expiring in the next 10 days.

It would be good to run this command regularly. The **cron** command can be set to execute **expchk** at regular intervals.

Chapter 6. Resigning a Zone

A zone needs to be re-signed when *any* change is made to it.

Resign the Zone with zonesigner

 # zonesigner -gends -zone zone-name zone-file output-file [ENTER]

Signing may take a few minutes to complete depending on the size of the zone file. This operation may appear to be unresponsive for a period of time, depending on the operating system's random number generator device. (See Chapter 2, *Configure DNSSEC-Tools* for more information on random number generators and DNSSEC-Tools.)

The output is a set of files outlined below.

Table 6.1. zonesigner Output Files

File	Description
`output-file.signed`	The signed zone file. The .signed is added by **zonesigner**.
`keyset-zone-name`	The keyset for the zone. This is stored in the directory specified by the configuration file and may have to be sent to the parent zone - see Chapter 7, *Creating a Signed Delegation - Child Zone Activity*.
`dsset-zone-name`	The dsset for the zone. This is stored in the directory specified by the configuration file and may have to be sent to the parent zone - see Chapter 7, *Creating a Signed Delegation - Child Zone Activity*.

Chapter 7. Creating a Signed Delegation - Child Zone Activity

This section describes the steps required by a child for creating a signed delegation.

Securely Transfer the Keyset to the Parent

If any of the zone's KSKs have changed since the last time this file was sent to the parent, then they keyset must also be transferred to the parent. If none of the zone's KSKs have changed, this step may be skipped.

Secure communication between the parent and child zone is done out-of-band.

Wait for the Parent to Publish the DS Record

Before proceeding, wait for the parent zone the publish the DS record. This may be found by using the **dig** command to retrieve the zone's DS record. The `aa` flag in the result must be set and the `ANSWER` section must not be empty.

You may continue if the DS record is the same as the value in the file generated in Chapter 3, *Initially Signing a Zone* or Chapter 6, *Resigning a Zone*.

 # dig @server-IP-address DS zone-name `[ENTER]`

```
; <<>> DiG 9.3.0 <<>> ... ... ;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 1,
ADDITIONAL:  0  ...  ;;ANSWER  SECTION:  zone-name  600  IN  DS  12960  5  1
5B10E822B935BC64DBEC2872A553EAA290443064 ; This value must match the data in your
dsset-zone-name file.
```

Chapter 8. Creating a Signed Delegation - Parent Zone Activity

This section describes the steps required by a parent for creating a signed delegation.

Ensure that the Child Keysets were Received Over a Secure Channel

Secure communication between the parent and child zone is done out-of-band.

Ensure that Each Received Keyset is for a Delegated Zone

The owner name for the DNSKEY record in the received keyset must correspond to a valid delegation.

 # grep DNSKEY keyset-child-zone-file [ENTER]

```
child-zone-name. 3600 IN DNSKEY 256 3 5 ( ... ); key id = keytag
```

`child-zone-name` must exist in the parent zone-file as a valid delegation.

 # grep NS zone-file [ENTER]

```
... child-zone-name NS server A ... ...
```

Re-sign the Parent Zone

Re-sign the parent zone using steps described in Chapter 6, *Resigning a Zone*.

Reload the Zone

The **rndc** command will reload the name server configuration files and zone contents. The name server process is assumed to be already running.

 # rndc reload zone-name [ENTER]

Chapter 9. Current ZSK Rollover (Pre-Publish Scheme)

Pre-Publish Rollover Scheme

This section gives the steps necessary for the Pre-Publish Rollover Scheme for ZSK rollover. The alternative, the double-signature method, is used for rolling over KSKs. Double signatures for records signed by the ZSK can increase the size of the zone many times. The Pre-Publish Rollover Scheme, although requiring more steps for the rollover, does not suffer from this problem. The size argument does not apply during KSK rollover since the DNSKEY RRset is the only record doubly signed by the KSK.

In the Pre-Publish Rollover Scheme, multiple ZSK keys are simultaneously maintained for a zone. These ZSKs are labeled the Current ZSK, the Published ZSK, and the New ZSK. The Current and Published ZSKs are used to sign the zone, while the New ZSK will be used in the future. When the Current ZSK expires, the following steps will be taken:

1. The Current ZSK becomes obsolete.
2. The Published ZSK becomes the Current ZSK.
3. The New ZSK becomes the Published ZSK.
4. A new New ZSK is generated.

A lot of record-keeping is required for managing a zone using the Pre-Publish Rollover Scheme. The DNSSEC-Tools utilities that automate ZSK rollover are described in Section 2. The actual steps taken in this rollover scheme are described in Section 3.

ZSK Rollover Using DNSSEC-Tools

The DNSSEC-Tools rollover commands simplify rollover to a great extent. A small amount of set-up is required, after which rollover happens automatically.

Gather Zone Data

The DNSSEC-Tools rollover commands can manage rollover of multiple zones. Zone files for these domains should be gathered into a single directory.

A number of zone parameters must be selected as well. These include such things as key length, number of ZSK keys to generate, and authentication algorithm. More information may be found in the man page for **zonesigner**. If these parameters will be used for every zone managed on this host, the DNSSEC-Tools configuration may be edited to have these values as the defaults.

Initial Signing of Zones

Using the **zonesigner** command, sign each zone with the parameters chosen for that zone. The resulting files should be left in place.

If the zone does no delegation, the following example command could be used. It will generate keys for the zone **example.com**, where the ZSK keys have a length of 1024, and then sign the zone with those keys.

 # zonesigner -genkeys -zsklength 1024 example.com [ENTER]

If the zone does delegation, the following example command could be used. It will generate keys for the zone **example.com**, where the ZSK keys have a length of 1024, and then sign the zone with those keys and generate DS records.

 # zonesigner -gends -genkeys -zsklength 1024 example.com [ENTER]

Create the Rollrec File

A *rollrec* file gives information to the DNSSEC-Tools rollover daemon about the zones it is managing. The **rollinit** command may be used to create a *rollrec* file for a number of zones at once, though the zones entries will all have the same type of data.

The following command will generate a *rollrec* file for two zones.

 # rollinit -o examples.rrf example1.com example2.com [ENTER]
 # cat examples.rrf
roll "example1.com"
zonefile "example1.com.signed"
keyrec "example1.com.krf"
curphase "0"
maxttl "0"
display "1"
phasestart "new"
roll "example2.com"
zonefile "example2.com.signed"
keyrec "example2.com.krf"
curphase "0"
maxttl "0"
display "1"
phasestart "new"
#

If different values are needed for different zones, **rollinit** may be used to generate entries for zones individually. The following commands will generate a *rollrec* file for two zones. The first **rollinit** command will use the default name for the signed zone file, while the second **rollinit** command will specify a non-default location for the signed zone file.

 # rollinit example1.com > examples.rrf
 # rollinit -zone signed-example2.com example2.com >> examples.rrf
 # cat examples.rrf
roll "example1.com"
zonefile "example1.com.signed"
keyrec "example1.com.krf"
curphase "0"
maxttl "0"
display "1"
phasestart "new"
roll "example2.com"
zonefile "signed-example2.com"
keyrec "example2.com.krf"
curphase "0"
maxttl "0"
display "1"
phasestart "new"
#

Run the DNSSEC-Tools Rollover Daemon

The DNSSEC-Tools rollover daemon is named **rollerd**. Using the *rollrec* file created in the previous step, **rollerd** will manage the rollover of a set of zones. This section describes how to manually start **rollerd**.

The following command will manually start *rollerd*. It is assumed that **rollerd** is started in the same directory that holds the *rollrec* file, *keyrec* files, zone files, and authentication keys created in previous steps. **rollerd** should be run as root.

```
# rollerd -dir . -logfile log-rollerd -loglevel info -rrf examples.rrf
#
```

See the **rollerd** man page for more information on **rollerd**'s options and execution.

Arranging for automatic execution of **rollerd** is operating system-dependent; as such, it is beyond the scope of this document.

Controlling the Rollover Process

The **rollerd** daemon can be controlled using the **rollctl** command. This command has a number of options that will modify **rollerd**'s operating parameters, such as the zones being managed (by changing the *rollrec* file), log level, and log file. It may also be used to start or stop a GUI interface to **rollerd** and to halt **rollerd**'s execution.

The following **rollctl** command retrieves status on each zone managed by **rollerd**. The zone name, roll/skip status, and rollover phase are displayed for each zone.

```
# rollctl -zonestatus
example1.com roll 0
example2.com roll 3
#
```

The following **rollctl** command starts a GUI interface to **rollerd**.

```
# rollctl -display
rollerd display started
#
```

The following **rollctl** command sets **rollerd**'s logging status to only record errors and fatal problems.

```
# rollctl -loglevel error
rollerd log level set to error
#
```

The following **rollctl** command changes the *rollrec* file in use by **rollerd**.

```
# rollctl -rollrec new.rrf
rollerd now using rollrec file new.rrf
#
```

The following **rollctl** command causes **rollerd** to stop execution.

```
# rollctl -halt
rollerd shutting down
#
```

Manual ZSK Rollover

The steps for performing a manual ZSK rollover are given Section 3.4 of [dnssec-operators-guide].

Chapter 10. KSK Rollover (Double-Signature Scheme)

This section gives the steps necessary for the double-signature scheme for KSK rollover. The alternative, the pre-publish method, is used for rolling over ZSKs. Double signatures for records signed by the ZSK can increase the size of the zone many times. The pre-publish scheme, although requiring more steps for the rollover, does not suffer from this problem. The size argument does not apply during KSK rollover since the DNSKEY RRset is the only record doubly signed by the KSK.

The DNSSEC-Tools utilities do not currently handle KSK rollover. The steps given below detail the double-signature scheme used for KSK rollover.

Manual KSK Rollover

The steps for performing a manual KSK rollover are given in Section 3.5 of [dnssec-operators-guide].

Chapter 11. Emergency ZSK Rollover (Current ZSK Compromise)

If the KSK is also compromised, perform the emergency KSK rollover first.

As long as there is a valid KSK signature over the ZSK, the KSK can continue to be used to inject false zone data. If both keys are compromised, clients are exposed to attacks on that data until the maximum of the expiration of the KSK's RRSIG (created by the ZSK) and the parent's signature over the DS of that KSK. (These attacks include signatures over false data, replay attacks of the old KSK, and replay attacks of the old DS.) Short TTLs allow recursive servers to more quickly recover from key-compromise situations, allowing them to get new keys more quickly. Key compromise exposes the secure recursive server to replays of the old key until the signature expires. The emergency procedures described for key rollover use the rationale that injection of valid but false data (which can be generated using the compromised key) is more serious than discontinuity in the ability to validate true data. Thus, during emergency ZSK rollover, there will be a period (up to twice the maximum zone TTL) where the cached zone data may not validate against the new ZSK. Also, the steps below are only useful if the Published and Current keys are kept separate from each other and if the Published ZSK has not also been compromised. If both ZSKs are compromised follow the steps in Chapter 13, *Emergency ZSK Rollover (Published and Current ZSK Compromise)* If only the Published key is compromised follow the steps in Chapter 12, *Emergency ZSK Rollover (Published ZSK Compromise)*.

Manual Emergency Current ZSK Rollover

The DNSSEC-Tools utilities do not currently handle emergency ZSK rollover. Section 6.2 of [dnssec-operators-guide] detail the pre-publish scheme used for ZSK rollover.

Chapter 12. Emergency ZSK Rollover (Published ZSK Compromise)

If the KSK is also compromised, perform the emergency KSK rollover first.

As long as there is a valid KSK signature over the ZSK, the KSK can continue to be used to inject false zone data. If both keys are compromised, clients are exposed to attacks on that data until the maximum of the expiration of the KSK's RRSIG (created by the ZSK) and the parent's signature over the DS of that KSK. (These attacks include signatures over false data, replay attacks of the old KSK, and replay attacks of the old DS.) Short TTLs allow recursive servers to more quickly recover from key-compromise situations, allowing them to get new keys more quickly. Key compromise exposes the secure recursive server to replays of the old key until the signature expires.

The emergency procedures described for key rollover uses that rationale that injection of valid but false data (which can be generated using the compromised key) is more serious than discontinuity in the ability to validate true data. Thus, during emergency ZSK rollover, there will be a period (up to twice the maximum zone TTL) where the cached zone data may not validate against the new ZSK.

Manual Emergency Published ZSK Rollover

The DNSSEC-Tools utilities do not currently handle emergency ZSK rollover. Section 6.3 of [dnssec-operators-guide] detail the pre-publish scheme used for ZSK rollover.

Chapter 13. Emergency ZSK Rollover (Published and Current ZSK Compromise)

If the KSK is also compromised, perform the emergency KSK rollover first.

The emergency procedures described for key rollover use the rationale that injection of valid but false data (which can be generated using the compromised key) is more serious than discontinuity in our ability to validate true data. Thus, during emergency ZSK rollover, there will be a period (up to twice the maximum zone TTL) where the cached zone data may not validate against the new ZSK.

The DNSSEC-Tools utilities do not currently handle emergency KSK rollover. However, the utilities may be used to automate *some* of the steps required.

Emergency Current and Published ZSK Rollover Using DNSSEC-Tools

The steps given below detail the steps that must be taken during emergency ZSK rollover when using DNSSEC-Tools to assist in rollover.

Stop Automatic Zone Rollover

The **rollerd** command must not be executing during this procedure.

 # rollctl -halt [ENTER]
#

Generate New Current and Published Keys

Creating new Current and Published ZSKs may be done with a single **zonesigner** execution.

 # zonesigner -genzsk zone.name [ENTER]
#

Fix the Keyrec File

The **zonesigner** command in the previous step will have left the compromised zone's keyrec file in an inconsistent state. Consequently, the keyrec file must be edited to return it to a valid state.

The steps below should be followed to fix the keyrec file for the Current ZSK keys.

1. Find the name of the zone's keyrec file. This may be done with the following command:

 # lsroll -keyrec -terse rollrec-file [ENTER]
#

2. Find the name of the zone's Current ZSK signing set. Look for the *zone* keyrec entry for the compromised zone, and find its *zskcur* entry. This holds the name of the Current ZSK signing set.

3. Get the names of the keys in the Current ZSK signing set. Look for the *set* keyrec entry for the Current ZSK signing set. The keys listed in that set's *keys* entry are the ZSK keys in the Current ZSK

signing set.

4. Edit the keyrec file and search for all *key* keyrec entries with a *keyrec_type* of "**zskcur**". Any keys with this type that are not in the Current signing set should be given the type "**zskobs**".

The steps below should be followed to fix the keyrec file for the Published KSK keys.

1. Find the name of the zone's keyrec file. This may be done with the following command:

 # lsroll -keyrec -terse rollrec-file [ENTER]
#

2. Find the name of the zone's Published signing set. Look for the *zone* keyrec entry for the compromised zone, and find its *zskpub* entry. This holds the name of the Published ZSK signing set.

3. Get the names of the keys in the Published signing set. Look for the *set* keyrec entry for the Published ZSK signing set. The keys listed in that set's *keys* entry are the ZSK keys in the Published ZSK signing set.

4. Edit the keyrec file and search for all *key* keyrec entries with a *keyrec_type* of "**zskpub**". Any keys with this type that are not in the Published signing set should be given the type "**zskobs**".

Reload the Zone

The **rndc** will reload the name server configuration files and the zone contents. The name server process is assumed to be already running.

 # rndc reload zone-name [ENTER]
#

Dispose of the Old Zone Key

Delete the old ZSK's *.private* and *.key* files.

Restart Automatic Zone Rollover

Automatic rollover may be restarted by executing the **rollerd** command. It should be given the same options as when it was originally started.

Manual Emergency Rollover of Current and Published ZSKs

Section 6.4 of [dnssec-operators-guide] detail the actions needed for emergency rollover of the Current and Published ZSKs.

Chapter 14. Emergency KSK Rollover (KSK Compromise)

The emergency procedures described for key roll-over use the rationale that injection of valid but false data (which can be generated using the compromised key) is more serious than discontinuity in our ability to validate true data. Thus, during emergency KSK roll-over, there will be a period (up to twice the maximum zone TTL) where it may not be possible to build an "authentication chain" from the zone data to the new KSK.

The DNSSEC-Tools utilities do not currently handle emergency KSK rollover. However, the utilities may be used to automate *some* of the steps required.

Emergency Current KSK Rollover Using DNSSEC-Tools

The steps given below detail the steps that must be taken during emergency KSK rollover when using DNSSEC-Tools to assist in rollover.

Inform Parent about the KSK Compromise

This communication between parent and child must be done securely using out-of-band mechanisms.

Wait for the Parent to Remove the Zone's DS Record

Before proceeding, wait for the parent zone to remove the DS record. This may be determined by using the **dig** command to retrieve the parent's DS record.

 # dig @parent-IP-address DS zone.name [ENTER]

```
...

:: flags: qr aa rd: QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL : 0

...

#
```

Stop Automatic Zone Rollover

The **rollerd** command must not be executing during this procedure.

 # rollctl -halt [ENTER]
#

Generate New Keys

Since the KSK has been compromised it must be regenerated. In addition, the ZSKs can no longer be trusted so they too must be regenerated. This may be done with a single **zonesigner** execution.

 # zonesigner -genkeys zone.name [ENTER]
#

Fix the Keyrec File

The **zonesigner** command in the previous step will have left the compromised zone's keyrec file in an inconsistent state. Consequently, the keyrec file must be edited to return it to a valid state.

The steps below should be followed to fix the keyrec file for the KSK keys.

1. Find the name of the zone's keyrec file. This may be done with the following command:

```
# lsroll -keyrec -terse rollrec-file [ENTER]
#
```

2. Find the name of the zone's Current KSK. Look for the *zone* keyrec entry for the compromised zone, and find its *ksk* entry. This holds the name of the KSK.

3. Edit the keyrec file and search for all *key* keyrec entries with a *keyrec_type* of "**ksk**". Any keys with this type that are not the Current KSK should be given the type "**kskobs**".

The steps below should be followed to fix the keyrec file for the Current ZSK keys.

1. Find the name of the zone's keyrec file. This may be done with the following command:

```
# lsroll -keyrec -terse rollrec-file [ENTER]
#
```

2. Find the name of the zone's Current ZSK signing set. Look for the *zone* keyrec entry for the compromised zone, and find its *zskcur* entry. This holds the name of the Current ZSK signing set.

3. Get the names of the keys in the Current ZSK signing set. Look for the *set* keyrec entry for the Current ZSK signing set. The keys listed in that set's *keys* entry are the ZSK keys in the Current ZSK signing set.

4. Edit the keyrec file and search for all *key* keyrec entries with a *keyrec_type* of "**zskcur**". Any keys with this type that are not in the Current signing set should be given the type "**zskobs**".

The steps below should be followed to fix the keyrec file for the Published KSK keys.

1. Find the name of the zone's keyrec file. This may be done with the following command:

```
# lsroll -keyrec -terse rollrec-file [ENTER]
#
```

2. Find the name of the zone's Published signing set. Look for the *zone* keyrec entry for the compromised zone, and find its *zskpub* entry. This holds the name of the Published ZSK signing set.

3. Get the names of the keys in the Published signing set. Look for the *set* keyrec entry for the Published ZSK signing set. The keys listed in that set's *keys* entry are the ZSK keys in the Published ZSK signing set.

4. Edit the keyrec file and search for all *key* keyrec entries with a *keyrec_type* of "**zskpub**". Any keys with this type that are not in the Published signing set should be given the type "**zskobs**".

Perform Child Activities

See Chapter 7, *Creating a Signed Delegation - Child Zone Activity* for the steps that need to be per-

formed if this zone is a secure delegation from another zone.

Reload the Zone

The **rndc** will reload the name server configuration files and the zone contents. The name server process is assumed to be already running.

 # rndc reload zone-name [ENTER]
#

Dispose of the Old Zone Key

Delete the old ZSK's *.private* and *.key* files.

Restart Automatic Zone Rollover

Automatic rollover may be restarted by executing the **rollerd** command. It should be given the same options as when it was originally started.

Manual Emergency Current KSK Rollover

Section 6.1 of [dnssec-operators-guide] details the manual steps that must be taken during emergency KSK rollover.

Chapter 15. Parent Action During Child KSK Compromise

During a KSK compromise the secure status of the child zone is dropped. This is done by deleting the DS record in the parent zone.

Ensure that the KSK Compromise Notification Came Over a Secure Channel

Authentication and communication between parent and child occurs out-of-band.

Delete the Child's Keyset File at the Parent

The DS record for the child should not be created. This can simply be achieved by removing the keyset file from the system.

Re-sign the Parent Zone

Re-sign the parent zone using steps described in Chapter 6, *Resigning a Zone*.

Reload the Zone

The **rndc** command will reload the name server configuration files and zone contents. The name server process is assumed to be already running.

 # rndc reload zone-name `[ENTER]`

Chapter 16. Migrate to the Toolset

The **zonesigner** tool simplifies the maintenance of a signed zone. It automates many of the routine tasks required for signing a zone. Given this, an operator already using BIND tools to maintain a signed zone may want to transition to **zonesigner**, while still retaining existing keys that are being used to sign a zone.

This section provides step-by-step instructions to transition from using BIND tools for maintaining a signed zone to using **zonesigner**. In the examples given below, the zone example.com is currently signed, signed zone file is maintained using **dnssec-signzone** command from BIND 9.3.1, and the following files are present:

Table 16.1. Example Files

File	Description
db-in.example.com.	Unsigned zone file
db-in.example.com..signed	Signed zone file
Kexample.com.+005+47670	KSK files prefix
Kexample.com.+005+48926	ZSK files prefix

Generate the Keyrec File

 # genkrf -zone=example.com -ksk=Kexample.com.+005+47670 -zskcur=Kexample.com.+005+48926db-in.example.com. db-in.example.com..signed

The **genkrf** command generates a keyrec file from existing key files. It also generates any additional keys that **zonesigner** uses. In the above example, **genkrf** will generate a new key `zskpub` along with the keyrec file named `example.com.krf`. It will display the following message if successful:

```
genkrf: file example.com.krf created successfully.
```

Verify the Keyrec File

Examine the contents of the keyrec file and ensure that the original KSK and ZSK files are being used.

 # grep Kexample.com.+005+47670 example.com.krf [ENTER]

```
kskdir "Kexample.com.+005+47670"
```

 # grep Kexample.com.+005+48296 example.com.krf [ENTER]

```
zskcur "Kexample.com.+005+48296"
```

Resign the Zone with zonesigner

See Chapter 6, *Resigning a Zone* for how to resign the zone.

Chapter 17. Configure a Secure Resolver

Introduction

This document has described how to configure and maintain a secure nameserver which supplies signed zones and delegations. All the signed zones and delegations within the scope of the server form an island of security from which nameserver data can be retrieved in a authenticated and verifiable way by a security aware resolver.

But there are times operationally when a recursing secure name server may need to refer to, and retrieve, data from servers outside this island of security. If the referral is to a non-secure name server there is no secure recourse and the chain of authentication is broken and this data can not then be trusted.

To extend the scope of security, a secure nameserver may be configured with public key data from other remote secure zones so that the chain of trust is expanded. The `trusted-keys` directive in the `named.conf` configuration file provides this capability.

The mechanism described below for extending the chain of trust should be used judiciously and comes with the added operational burden of verifying and maintaining key validity and timeliness.

The following is an example of a `trusted-keys` directive in a `named.conf` which provides verification of data retrieved from the `se.` and `dnssec-tools.org.` zones.

Note: Key data may be different from that shown and should be obtained as described below.

```
trusted-keys {

se.     257     3     5     "AwEAAaxPMcR2x0HbQV4WeZB6oEDX+r0QM65KbhTjrW1ZaARmPhEZZe3Y
9ifgEuq7vZ/zGZUdEGNWy+JZzus0lUptwgjGwhUS1558Hb4JKUbbOTcM
8pwXlj0EiX3oDFVmjHO444gLkBOUKUf/mC7HvfwYH/Be22GnClrinKJp
1Og4ywzO9WglMk7jbfW33gUKvirTHr25GL7STQUzBb5Usxt8lgnyTUHs
1t3JwCY5hKZ6CqFxmAVZP20igTixin/1LcrgX/KMEGd/buvF4qJCydui
eHukuY3H4XMAcR+xia2nIUPvm/oyWR8BW/hWdzOvnSCThlHf3xiYleDb t/o1OTQ09A0=";

dnssec-tools.org. 257 3 5 "AQOoEFn3VnV1qDwnNX9GlukAsbL7buCk6Wmt3VG9BOVae84VVc/yWghg
tFM/WKw/5243XoBEeNyaahRIrlAJEnErLUWlKO/YuWkasRN4jkS2dDjS
MWgjdGxzux+e0UV2UZfpjyygYvaD9U8xTwwzLYLDkamr1SCaHWCWUOO+                    QMa/
WY//r3ObbOFOYCvyqvsLRwofSFnQnsbihKbcP9HQSDQ4iRqbCTMV
B+yq5NXiFoZT05Sqm/ijOrjLznZkUqIal9EXqyhNT0dTa9Gdn8+tfn+l                    YAp-
wK91NA2YG/3t8ZKTYjDLe1YlwKg8OBTTN4ARap+265EtE87BhE6ZK fp+DUx4N";

};
```

The format of the directive is:

```
trusted-keys { <zone> <flags> <protocol> <algorithm> <quoted-key-string>; };
```

The *`flags`*, *`protocol`*, *`algorithm`* and *`quoted-key-string`* data may be obtained using the following **dig** command, but the content of the string should be verified in a secure out-of-band way to ensure its validity.

 # dig se. DNSKEY

```
;; Truncated, retrying in TCP mode. ; <<>> DiG 9.3.1 <<>> se. DNSKEY ;; global op-
```

```
tions: printcmd ;; Got answer: ;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id:
31166 ;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 8, ADDITIONAL: 0 ;; QUES-
TION SECTION: ;se. IN DNSKEY ;; ANSWER SECTION: se. 3600 IN DNSKEY 257 3 5 AwEAAaxP-
McR2x0HbQV4WeZB6oEDX+r0QM65KbhTjrW1ZaARmPhEZZe3Y
9ifgEuq7vZ/zGZUdEGNWy+JZzus0lUptwgjGwhUS1558Hb4JKUbbOTcM
8pwXlj0EiX3oDFVmjHO444gLkBOUKUf/mC7HvfwYH/Be22GnClrinKJp
1Og4ywzO9WglMk7jbfW33gUKvirTHr25GL7STQUzBb5Usxt8lgnyTUHs
1t3JwCY5hKZ6CqFxmAVZP20igTixin/1LcrgX/KMEGd/buvF4qJCydui
eHukuY3H4XMAcR+xia2nIUPvm/oyWR8BW/hWdzOvnSCThlHf3xiYleDb t/o1OTQ09A0= ...
```

Note: from the output select the DNSKEY whose flags have the zone signing key bit set (257).

Once the 'named.conf' is edited as above, the configuration can be reloaded with:

 # rndc reload

It may also be necessary to flush the cache data before retrieving authenticated results:

 # rndc flush

To verify that the `trusted-keys` directive is working properly perform a secure **dig** at the configured server for the remote signed zone data and observe that the `ad` flag is set in the response. For example:

 # dig @localhost se. ANY +dnssec

```
;; Truncated, retrying in TCP mode. ;; Connection to ::1#53(::1) for se. failed:
connection refused. ; <<>> DiG 9.3.1 <<>> @localhost ANY se. +dnssec ; (2 servers
found) ;; global options: printcmd ;; Got answer: ;; ->>HEADER<<- opcode: QUERY,
status: NOERROR, id: 56473 ;; flags: qr rd ra ad; QUERY: 1, ANSWER: 23, AUTHORITY:
9, ADDITIONAL: 1 ...
```

References

[dnssec-operators-guide] SPARTA, Inc.. *Step-by-Step DNS Security Operator Guidance Document*. 31 August 2005.

[dnssec-operational-practices] Olaf Kolkman and Miek Gieben. *DNSSEC Operational Practices*. 24 October 2005. draft-ietf-dnsop-dnssec-operational-practices-06.txt (work in progress).