Lab 10.1 – Implementing DNSSEC

Objective:
Deploy DNSSEC-signed zones.

Background
DNSSEC (or DNS Security Extensions) provide security to the zone files.

Note:
In the steps below, we are using
myzone.net - our domain
db.myzone.net – zonefile for the domain
Kmyzone.net.+005+12345.key/private = ZSK generated
Kmyzone.net.+005+67890.key/private = KSK generated

Steps:

A. **DNSSEC Validation.** To allow your recursive DNS servers to validate DNSSEC-signed zones.

1. Update the DNS configuration. Add options in the configuration file named.conf to allow DNSSEC.

   These options must be enabled:

   ```
   dnssec-enable yes;
   dnssec-validation yes|auto;
   ```

   `dnssec-enable` allows named to respond to DNS requests from DNSSEC-aware clients. The default is yes, but is best added in the named.conf so you know how to turn it off.

   If `dnssec-validation` is set to auto, it defaults to the DNS root zone as the trust anchor.

   If set to yes, a trust anchor must be explicitly configured using the managed-keys or trusted-keys option.

   ```
   managed-keys {
      // root key
      ". Initial-key 257 3 3 "<key-here>"
   };

   trusted-keys {
      // parent zone
      <myzone.net> 257 3 5 "<key-here>";
   };
   ```

2. Using the dig command, do a lookup for a dnssec-enabled domain. The output should show an RRSIG next to the record you asked.

```
dig @nameserver +dnssec +multiline www.apnic.net
```

Also check for the AD bit in the message header flags. It should look something like:

```
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40679
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
```

B. **Signing the zone**.

1. Generate the key pair.

This command generates the ZSK.

```
dnssec-keygen –r /dev/random –a <algorithm> –b <keysize> \
–n ZONE <myzone>
```

ex:

```
dnssec-keygen –r /dev/random –a rsasha1 –b1024 –n ZONE myzone.net
```

The defaults are RSASHA1 for the algorithm, with 1024 bits for ZSK and 2048 bits for KSK. Since these are all defaults, we can just issue the command:

```
dnssec-keygen –r /dev/random <myzone>
```

This command generates the KSK

```
dnssec-keygen –a <algorithm> –b <keysize> –f KSK –n ZONE <myzone>
```

Or simply

```
dnssec-keygen –f KSK <myzone>
```

This generates 4 files.

2. Include the public DNSKEYs in the zone file.

You can either copy the entire file or reference to it using the $INCLUDE directive. To do the latter, simply add the lines below. Note that you are including only the public portion (.key) into the zone file. The private portion (.private) must be kept secure.

```
$INCLUDE "K<myzone>.+005+<id_of_zsk>.key"
```

```
$INCLUDE "K<myzone>.+005+<id_of_ksk>.key"
```

3. Sign the zone using the secret keys. The syntax is:

   dnssec-signzone [options] {zonefile} [key…]

   ```
   dnssec-signzone –o <zonename> –N INCREMENT -f <output-file> -t \
   –k <KSKfile> <zonefile> <ZSKfile>
   ```

   ex:

   ```
   dnssec-signzone –o myzone.net –N INCREMENT –f <output-file> -t -k \
   Kmyzone.net.+005+12345 db.myzone.net Kmyzone.net.+005+67890
   ```

   If **–f** is not specified, the output file will append a `.signed` in the zonefile.

   <myzone>.signed

   Smart Signing

   You can use the **–S** option so that keys will be imported into the zone automatically. Specify a keys repository in named.conf, which will be checked by named when executed.

   ```
   options {
      keys-directory { "/path/to/keys";
      };
   ```

   Then issue the command:

   ```
   dnssec-signzone –S db.myzone.net
   ```

   The output file is bigger than the original zone file. Check with the commands `ls –al` or `wc`.

C. NSEC and NSEC3.

NSEC records are created to prove the non-existence of a record. It builds a linked list of all the records in the zone file. The problem with this is it allows anyone to list the zone content. This is called "zone walking." Some tools, like the `ldns-walk` (included in the LDNS library), can be used to do exactly this.

NSEC3 can be used to provide more security. It uses a hashing algorithm to output a "hash" to replace the real domain names. This makes it difficult for an attacker, but not totally impossible.

1. Using NSEC3 to generate keys.

   To do this, you may use NSEC3RSASHA1 as your algorithm. The easier way to do this is to use -3 option instead. This option allows a few other algorithms such as RSASHA256 and RSAHSHA512 but sets NSEC3RSASHA1 as default.

```
dnssec-keygen –r /dev/random -3 <myzone>
dnssec-keygen –f ksk –r /dev/random -3 <myzone>
```

2. Sign the zone with a salt.

```
dnssec-signzone –A -3 <salt> –o <zonename> –N INCREMENT -f <output-file> -t –k <KSKfile> <zonefile> <ZSKfile>
```

   The salt is a random hexadecimal number appended to the domain before hashing. It's a public data that is part of the NSEC3PARAM record. It must be changed once in a while or on regular intervals.

   To generate the salt, you can use either of these:

```
date | sha1sum | cut –b 1-16
head –c 1000 /dev/random | sha1sum | cut –b 1-16
```

   Example:

```
dnssec-signzone –A -3 $(head –c 1000 /dev/random | sha1sum | cut –b \
1-16) –o myzone.net –N INCREMENT –f <output-file> -t -k \
Kmyzone.net.+005+12345 db.myzone.net Kmyzone.net.+005+67890
```

   To use NSEC3 without a salt, simply use a single dash.

D. Publishing the zone.

1. Reconfigure to load the signed zone. Edit named.conf and point to the signed zone. For example:

```
zone "<myzone>" {
      type master;
      # file "db.myzone.net";
      file "db.myzone.net.signed";
};
```

   Change the file to point to the signed zone.

2. Push the DS record up to your parent domain. Another output of the dnssec-signzone command is the file dsset-<yourdomain> (ex: dsset-myzone.net) which contains the DS records.

   The contents of the file look something like this:

```
myzone.net.          IN DS 4297 5 1 C5A8C518B2208463F87CB30E35F247DD7EACCDB1
myzone.net.          IN DS 4297 5 2 27E89E4A769F6C6BC889BB6F2E98374CA835D2B8C750D5505F32144E 1E79B881
```

   where:

   Keytag = 4297
   Algorithm = 5
   Digest type = 1 (for the first line), 2 (for the second line)
   Digest = `C5A8C518B2208463F87CB30E35F247DD7EACCDB1`

   You must contact the parent zone to communicate these values to them. This can be done by sending this file, or filling up an online form provided by your parent zone (or domain registrar). In the class, send/copy the file using SCP.

   The parent zone will then include the DS record in their zonefile. The $INCLUDE statement can be used at this stage.

```
$INCLUDE "dsset-myzone.net."
```

   You may check if it has been successfully added using dig.

```
dig @nameserver +noadditional DS myzone.net | grep DS
```

3. For slave servers, the configuration is simple.

   In the configuration file, add in the options section:

```
dnssec-enable yes;

dnssec-validation auto | yes;
```

   Then edit the zone section to point to the new signed zonefile. After reload, verify that this file exists in the folder specified in the config.

```
zone "<myzone>" {
      type slave;
      masters { X.X.X.X; };
       # file "db.myzone.net";
      file "db.myzone.net.signed";
};
```

4. Check if DNSSEC is working using the dig command.

```
dig @localhost +dnssec +multiline myzone.net
```