

THE UNITED REPUBLIC OF TANZANIA
TANZANIA COMMUNICATIONS REGULATORY AUTHORITY
ISO 9001:2015 CERTIFIED



Tanzania Computer Emergency Response Team (TZ-CERT)

**Domain Name System Security Extension (DNSSEC) Deployment
Guideline**

Issued by:

The Director General

Tanzania Communications Regulatory Authority
Mawasiliano Towers
20 Sam Nujoma Road
P.O Box 474
14414 DAR ES SALAAM
TANZANIA

November, 2019

ABSTRACT

Domain Name System abbreviated as DNS, is a part of the global Internet infrastructure that translates domain names into Internet Protocol (IP) addresses. It is a standard protocol that helps Internet users access information from websites through use of human readable addresses known as domain names. Similar to the telephone directory, which lets users look up the name of a person and discover their number, DNS lets internet users type the address of a website and automatically locate its Internet Protocol (IP) address.

Unlike computers, humans are incapable to remember dozens of IP addresses when browsing web pages instead they need easy-to-remember names to easily locate computing resources on the internet. It is because of this, DNS was invented to foster internet usability.

Apart from translating names into numerical numbers and vice versa, DNS also plays an important role in email communications, online phone services, and Virtual Private Network (VPN) connections. Its ability to provide a distributed directory service across the globe, makes it an essential component of internet functionality. Despite that, it's often poorly secured or improperly configured making it vulnerable to cyber threats. Because of this, majority of individuals and organizations worldwide have been encountering a range of attacks such as DNS hijacking, in which DNS queries are incorrectly resolved in order to redirect users to malicious sites.

It is essential to put in place effective mechanism to enhance safety of electronic communications in line with the section 6(i) of the Electronic and Postal Communications (*Computer Emergency Response Team*) Regulation, 2018. As part of the mechanism, this guideline has been developed for TZ-CERT constituencies and stakeholders on deployment of **Domain Name System Security Extensions (DNSSEC)**, a suite of extensions that add security to the DNS protocol by validating DNS responses. When DNSSEC is enabled, DNS responses are digitally signed to validate their authenticity. This prevents attackers from forging the signature to misdirect users to malicious websites.

This guideline applies to all organizations operating or managing DNS systems whether for enterprise or commercial use. It is envisaged that every DNS holder will implement DNSSEC to promote safe and secure functioning of the internet in Tanzanian cyberspace.

TABLE OF CONTENTS

1. PURPOSE AND SCOPE.....	3
2. AUDIENCE.....	3
3. DEFINITIONS	3
4. INTRODUCTION	4
4.1. Domain Name System (DNS)	4
4.3. DNS attacks	6
4.4. The DNS Security Extensions (DNSSEC).....	7
4.4.1. How DNSSEC works	7
4.4.2. Advantages of DNSSEC	8
5. DNSSEC DEPLOYMENT PROCEDURES	9
5.1. DNSEC Deployment for Ubuntu Server.....	9
5.1.1. Steps to deploy DNSSEC for Ubuntu.....	9
5.1.2. Configure DNSKEY records with the registrar	16
5.2. DNSSEC Deployment for Windows.....	16
5. CONCLUSION	26
6. REFERENCES.....	26

1. PURPOSE AND SCOPE

This guideline assists TZ-CERT constituencies and stakeholders in understanding deployment of Domain Name System Security Extension (DNSSEC) in order to enhance security of DNS. It covers deployment of DNSSEC running on both Windows and Ubuntu Servers operating systems.

2. AUDIENCE

The targeted audience for this guideline are System Administrators, Network Operators and related personnel responsible to manage Domain Name System (DNS).

3. DEFINITIONS

This section defines a number of terms used in this document.

a) **Top-Level Domain (TLD)**

These are domains at the highest level in the hierarchical Domain Name System of the Internet installed in the root zone of the name space.

b) **DNS resolver**

The first stop in the DNS lookup, it is responsible for dealing with the client that made the initial request. The resolver starts the sequence of queries that ultimately leads to a URL being translated into the necessary IP address.

c) **Root nameserver**

The nameserver the first step in translating (*resolving*) human readable host names into IP addresses. It can be thought like an index in a library that points to different racks of books - typically it serves as a reference to other more specific locations.

d) **Top Level Domain server (TLD) nameserver**

The nameserver next step in the search for a specific IP address, and it hosts the last portion of a hostname (*e.g. abc.go.tz, the TLD server is "com"*).

e) **Authoritative nameserver**

The final nameserver the last stop in the nameserver query. If the authoritative name server has access to the requested record, it will return the IP address for the requested hostname back to the DNS Recursor (*the librarian*) that made the initial request.

f) **Recursive query**

In a recursive query, a DNS client request the DNS server (*typically a DNS recursive resolver*) and will respond to the client with either the requested resource record or an error message if the resolver can't find the record.

g) **Iterative query**

The DNS client will allow a DNS server to return the best answer it can. If the queried DNS server does not have a match for the query name, it will return a referral to a DNS server authoritative for a lower level of the domain namespace. The DNS client will then make a query to the referral address. This process continues with additional DNS servers down the query chain until either an error or timeout occurs.

h) **Delegation Signer (DS)**

The DNSSEC record type that is used to secure a delegation. DS records are used to build authentication chains to child zones.

i) **DNSKEY resource records**

This stores a public cryptographic key for verifying a signature. The DNSKEY record is used by a DNS server during the validation process. It can store public keys for a zone signing key (ZSK) or a key signing key (KSK).

j) **RRSIG**

Stands for Resource Record Signature, it holds a DNSSEC signature for a record set.

k) **Next Secure (NSEC)**

An NSEC record is used to prove nonexistence of a DNS name. NSEC records prevent *spoofing attacks* that are intended to fool a DNS client into believing that a DNS name does not exist.

l) **Next Secure 3 (NSEC3)**

NSEC3 is a replacement or alternative to NSEC that has the additional benefit of preventing "*zone walking*" which is the process of repeating NSEC queries in order to retrieve all the names in a zone. A DNS server running Windows Server 2012 or a later operating system supports both NSEC and NSEC3. A zone can be signed with either NSEC or NSEC3, but not both.

4. INTRODUCTION

4.1. Domain Name System (DNS)

The Domain Name System (DNS) is a central part of the Internet, translating domain names (such as *tzcert.go.tz*) into numeric internet addresses (such as 196.41.76.33) needed by servers, routers, and other network devices to route traffic across the internet to its proper destination.

Internet is an IP network. Using the Internet on any device starts with the DNS. Any device connected to the internet has a unique public IP address that must be known to any other host willing to communicate with. On the contrary, it would be impossible for a human being to remember all the IP addresses it intends use on the Internet. DNS eliminates the need for humans to memorize IP addresses.

4.2.How DNS works?

The following steps describe how DNS works: -

- a) On typing a web page such as “*tzcert.go.tz*” on the browser a query is received by a DNS recursive resolver, which can be operated by either Internet Service Provider (ISP), wireless carrier or a third party.
- b) The resolver then queries a DNS root nameserver “.”.
- c) The root server then responds to the resolver with the address of a Top Level Domain (TLD) DNS server (such as .com or .net), which stores the information for its domains. For the case of “*tcra.go.tz*”, the request point to the “.tz” TLD.
- d) The resolver then makes a request to the “.tz” TLD.
- e) The TLD server then responds with the IP address of the domain’s nameserver i.e. “*tcra.go.tz*”.
- f) Lastly, the recursive resolver sends a query to the domain’s nameserver.
- g) The IP address for “*tcra.go.tz*” returned to the resolver from the nameserver.
- h) The DNS resolver responds to the web browser with the IP address of the domain requested initially.
Once DNS lookup have returned the IP address for “tcra.go.tz”, the browser is able to make the request for the web page:
- i) The browser makes a Hypertext Transfer Protocol (HTTP) request to the IP address.
- j) The server at that IP address returns the webpage to be rendered in the browser (step i).

N.B: This whole process might seem complicated, but takes very little time.

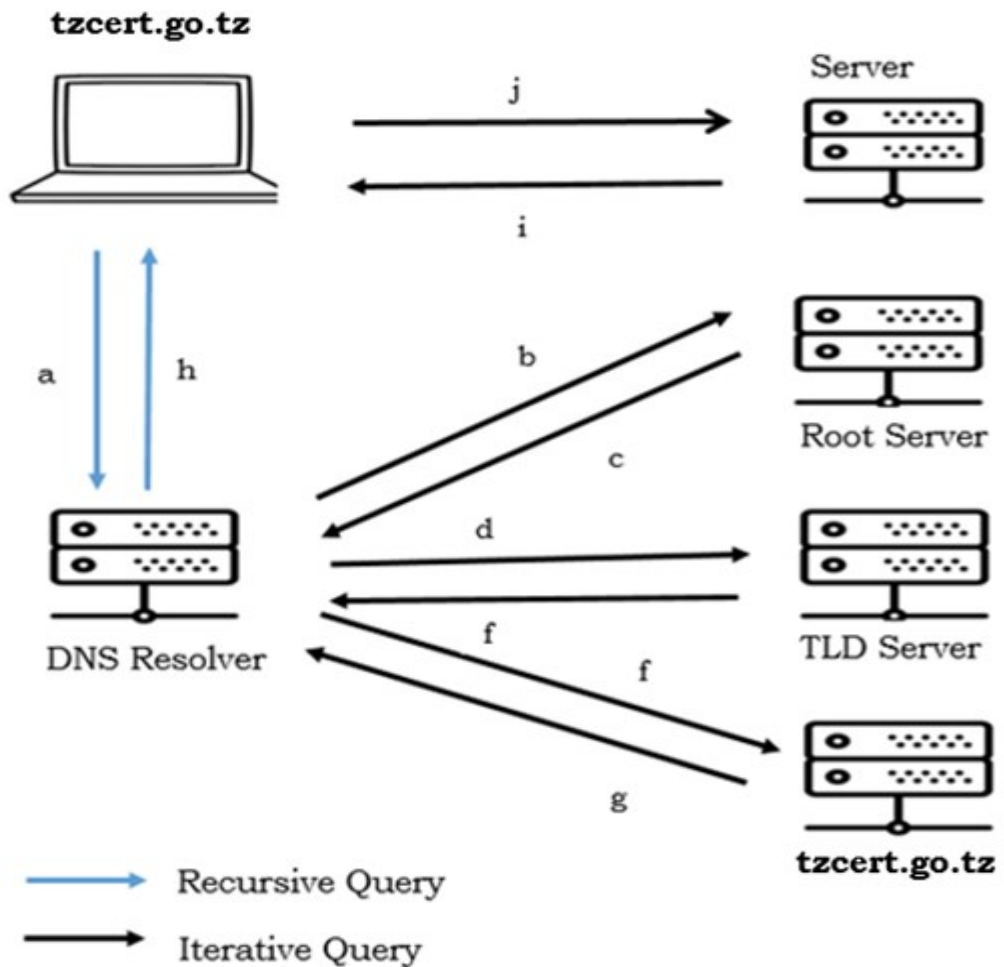


Figure 1: Complete DNS lookup and webpage query

4.3.DNS attacks

DNS by itself is not secure. Like many internet protocols, it was not designed with security in mind and thus contains vulnerabilities. The reason lays on the fact that, when a recursive resolver sends a query to an authoritative name server, the resolver has no way to verify the authenticity of the response. The resolver can only check that a response appears to come from the same IP address where the resolver sent the original query.

However, relying on the source IP address of a response alone is not a secure mechanism for authentication, since the source IP address of a DNS response packet can be easily *forged*, or *spoofed*. In this case, the DNS resolver cannot easily detect a forged response to one of its queries. An attacker can easily pretend to be as the authoritative server that a resolver originally queried by spoofing a response that appears to come from that authoritative server. In other words, an attacker can redirect a web request to a potentially malicious site without user’s knowledge.

This security flaw, coupled with advances in technology, makes the DNS vulnerable to cyber threats. Attackers have found a number of ways to target and exploit DNS servers, some of the most common ones are: -

a) **DNS spoofing/cache poisoning**

The attack whereby forged DNS data is introduced into a DNS resolver's cache, resulting in the resolver returning an incorrect IP address for a domain. Instead of going to the correct website, traffic can be diverted to a malicious machine or anywhere else the attacker desires; often this will be a replica of the original site used for malicious purposes such as distributing malware or collecting login credentials.

b) **DNS tunneling**

The attacker uses other protocols to tunnel through DNS queries and responses. Attackers can use SSH, TCP, or HTTP to pass malware or stolen information into DNS queries goes undetected by most firewalls.

c) **DNS hijacking**

In DNS hijacking the attacker redirects queries to a different domain name server. This can be done either with malware or with the unauthorized modification of a DNS server. Although the result is similar to that of DNS spoofing, this is a fundamentally different attack targeting the DNS record of the website on the nameserver rather than a resolver's cache.

4.4.The DNS Security Extensions (DNSSEC)

In attempt to mitigate security limitations of DNS, the Domain Name System Security Extensions (DNSSEC) was introduced, a security mechanism to protect DNS against cyber-threats by digitally signing data to help ensure its validity.

4.4.1. How DNSSEC works

The original purpose of DNSSEC was to protect Internet clients from counterfeit DNS data by verifying digital signatures embedded in the data. It strengthens authentication mechanism in DNS using **digital signatures** based on **public key cryptography**. The following is a brief description of DNSSEC functioning: -

- a) When a visitor enters the domain name in a browser, the resolver verifies the digital signature.
- b) If the digital signatures in the data match those that are stored in the master DNS servers, then the data is allowed to access the client computer making the request.

- c) The DNSSEC digital signature ensures that you are communicating with the site or Internet location you intended to visit.
- d) DNSSEC uses a system of public keys and digital signatures to verify data. It simply adds new records to DNS alongside existing records. These new record types, such as RRSIG and DNSKEY, can be retrieved in the same way as common records such as A, CNAME and MX.
- e) These new records are used to digitally "sign" a domain, using a method known as public key cryptography.
- f) A signed nameserver has a public and private key for each zone. When someone makes a request, it sends information signed with its private key; the recipient then unlocks it with the public key. If a third party tries to send untrustworthy information, it won't unlock properly with the public key, so the recipient will know the information is bogus.

Essentially, DNSSEC adds below important features to the DNS protocol: -

- a) **Data origin authentication**, allows a resolver to cryptographically verify that the data it received actually came from the zone where it believes the data originated.
- b) **Data integrity protection**, allows the resolver to know that the data hasn't been modified in transit since it was originally signed by the zone owner with the zone's private key.
- c) **Authenticated denial of existence**, allows a resolver to validate that a certain domain name does not exist. It is also used to signal that a domain name exists but does not have the specific resource record (RR) type it was asking for.

NB: DNSSEC does not provide data confidentiality because it does not include encryption algorithms. It only carries the keys required to authenticate DNS data validity. Furthermore, it does not protect DNS server against Denial of Service Attacks.

4.4.2. Advantages of DNSSEC

To sum up, DNSSES offers the following advantages: -

- a) Strengthens trust in the Internet by helping to protect users from redirection to fraudulent websites and unintended addresses. In such a way, malicious activities like cache poisoning, pharming, and man-in-the-middle attacks can be prevented.
- b) Authenticates the resolution of IP addresses with a cryptographic signature, to make sure that answers provided by the DNS server are valid and authentic. In case DNSSEC is properly enabled for your

domain name, the visitors can be ensured that they are connecting to the actual website corresponding to a particular domain name.

5. DNSSEC DEPLOYMENT PROCEDURES

5.1.DNSEC Deployment for Ubuntu Server

Assumptions – The DNS is running on Ubuntu 16.04 and bind9 for master and slave authoritative nameserver. In case your DNS is not locally hosted, please ask the hosting provider to undertake DNSSEC deployment on your behalf.

Table No.1: Setup environment

1. Domain Name: abc.go.tz <i>(For demo purposes, it should be replaced by actual domain name)</i>
2. Master Nameserver: a) IP Address: 1.1.1.1 b) Hostname: master.abc.go.tz c) OS: Ubuntu 16.04
3. Slave Nameserver: a) IP Address: 2.2.2.2 b) Hostname: slave.abc.go.tz c) OS: Ubuntu 16.04
4. File location and names: <i>(The names and locations of configuration and zone files of BIND different according to the Linux distribution used)</i>
5. For Debian/Ubuntu <ul style="list-style-type: none">• Service name: <i>bind9</i>• Main configuration file: <i>/etc/bind/named.conf.options</i>• Zone names file: <i>/etc/bind/named.conf.local</i>• Default zone file location: <i>/var/cache/bind/</i>

5.1.1. Steps to deploy DNSSEC for Ubuntu

A. DNSSEC Master Server Configuration

- i) Enable DNSSEC by adding the following configuration directives inside options {}

```
nano /etc/bind/named.conf.options
```

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

NOTE: It is possible that these are already added in some distributions. Navigate to the location of your zone files.

```
cd /var/cache/bind
```

- ii) Create a Zone Signing Key(ZSK) with the following command.

```
dnssec-keygen -a <ALGORITHM> -b <BITS> -n ZONE <ZONENAME>
```

- iii) Replace **ALGORITHM**, **BITS**, and **ZONENAME** . If not specified, the default values are RSASHA1 for the algorithm (-a), and a keysize (-b) of 1024 for ZSK and 2048 for KSK.

Example:-

```
dnssec-keygen -a NSEC3RSASHA1 -b 2048 -n ZONE abc.go.tz
```

NOTE: If you have installed **haveged**, it'll take only a few seconds for this key to be generated; otherwise it'll take a very long time. **haveged** is a userspace entropy daemon which is not dependent upon the standard mechanisms for harvesting randomness for the system entropy pool.

Sample output:

```
root@master:/var/cache/bind# dnssec-keygen -a NSEC3RSASHA1 -b  
2048 -n ZONE abc.go.tz  
Generating key pair.....+++ .....+++  
Kabc.go.tz.+007+05190
```

- iv) Create a Key Signing Key(KSK) with the following command.

```
dnssec-keygen -f KSK -a NSEC3RSASHA1 -b 4096 -n ZONE abc.go.tz
```

Sample output:

```
root@master:/var/cache/bind# dnssec-keygen -f KSK -a
NSEC3RSASHA1 -b 4096 -n ZONE abc.go.tz
Generating key
pair.....++
Kabc.go.tz.+007+35837
```

NOTE: The directory will now have 4 keys - private/public pairs of ZSK and KSK.

ZONES SIGNING:

a) Manual signing

We have to add the public keys which contain the **DNSKEY** record to the zone file. The following for loop will do this.

```
for key in `ls Kabc.go.tz*.key`
do
echo "\$INCLUDE $key">> abc.go.tz.zone
done
```

v) Sign the zone with the “dnssec-signzone” command.

```
dnssec-signzone -3 <salt> -A -N INCREMENT -o <zonename> -t
<zonefilename>
```

vi) Replace salt with something random. Here is an example with the output.

```
root@master:/var/cache/bind# dnssec-signzone -A -3 $(head -c 1000
/dev/random | sha1sum | cut -b 1-16) -N INCREMENT -o abc.go.tz -t
abc.go.tz.zone
```

Verifying the zone using the following algorithms: NSEC3RSASHA1.
Zone signing complete:

Algorithm: NSEC3RSASHA1: KSKs: 1 active, 0 stand-by, 0 revoked
ZSKs: 1 active, 0 stand-by, 0 revoked

abc.go.tz.zone.signed

Signatures generated: 14

Signatures retained: 0

Signatures dropped: 0

Signatures successfully verified: 0

```
Signatures unsuccessfully verified:      0
Signing time in seconds:                 0.046
Signatures per second:                   298.31
Runtime in seconds:                      0.056
```

- vi) Enter “A 16” character string as the “salt”. The following command outputs a random string of 16 characters which will be used as the salt.

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

This creates a new file named “**abc.go.tz.zone.signed**” which contains **RRSIG** records for each DNS record.

- vii) Configure BIND to load this “signed” zone with following command:-

```
nano /etc/bind/named.conf.local
```

- viii) Change the file option inside the zone {} section.

```
zone "abc.go.tz" IN {
    type master;
    file "abc.go.tz.signed";
    allow-transfer { 2.2.2.2; };
    allow-update { none; };
};
```

- ix) Save this file and reload bind

```
service bind9 reload
```

- x) Check if for the DNSKEY record using dig on the same server by typing the following command.

```
dig DNSKEY abc.go.tz. @localhost +multiline
```

Sample output:

```
root@master:/var/cache/bind# dig DNSKEY abc.go.tz. @localhost
+multiline
;; Truncated, retrying in TCP mode.
```

```

; <<>> DiG 9.8.4-rpz2+rl005.12-P1 <<>> DNSKEY abc.go.tz. @localhost
+multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43986
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL:
0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;abc.go.tz.      IN DNSKEY

;; ANSWER SECTION:
abc.go.tz.      86400 IN DNSKEY  256 3 7 (
  AwEAAActPMYurNEyhUgHjPctbLCI1VuSj3xcjI8QFTpdM
  8k3cYrFWB/WLNKjnnjt98nPmHv6frnuvs2LKIvGzz++
  kVwVc8uMLVyLOxVeKhygDurFQpLNNdPumuc2MMRvV9me
  fPrdKWtEEtOxq6Pce3DW2qRLjyE1n1oEq44gixn6hjgo
  sG2FzV4fTQdxYcZlYjsaZwy0Kww4HpIaozGNjODQVI/
  f3JtLpE1MYEb9DiUVMjkwVR5yH2UhJwZH6VVvDOZg6u6
  YPOSUDVvyofCGcICLqUOG+qITYVucyIWgZtHZUb49dpG
  aJTAdVKlOTbYV9sbmHNUmuGt+1/rc+StsjTPTHU=
) ; key id = 40400
abc.go.tz.      86400 IN DNSKEY  257 3 7 (
  AwEAAa2BE0dAvMs0pe2f+D6HaCyiFSHw47BA82YGs7Sj
  qSqH3MprNra9/4S0aV6SSqHM3iYzt5NRQNTNTRzkE18e
  3j9AGV8JA+xbEow74n0eu33phoxq7rOpd/N1GpCrXUsG
  kK4PDkm+R0hhfufe1ZOSoiZUV7y8OVGFB+cmaVb7sYqB
  RxeWPi1Z6Fj1/5oKwB6Zqbs7s7pml/GcjTvdQkMFtOQ
  AFGqaaSxVrisjq7H3nUj4hJIJ+SStZ59qfW3rO7+Eqgo
  1aDYaz+jFHZ+nTc/os4Z51eMWSZPYRnPRJG2EjJmkBrJ
  huZ9x0qnejEjUPAcUgMVqTo3hkRv0D24I10LAVQLEtuw/
  QOuWMG1VjybzLbXi5YScwcBDAGtEpsQA9o7u6VC00DGh
  +2+4RmgrQ7mQ5A9MwhglVPaNXKuI6sEGlWripGTwm425
  JFv2tGHROS55Hxx06A416MtxBpSEaPMYUs6jSlYf9cJB
  BMV24OjkCxdz29zi+OyUyHwirW51BFSaOQuzaRiOsovM
  NSEgKWLwzwsQ5cVJBEMw89c2V0sHa4yuI5rr79msRgZT
  KCD7wa1Hyp7s/r+ylHhjprZwViOPU7tAGZ3IkkJ2SMI
  e/h+FGiwXXhr769EHbVE/PqvdbpcsgsDqFu0K2oqY70u
  SxnsLB8uVKYlZjG+UloQzefBluQl
) ; key id = 62910

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Wed Nov 27 18:18:30 2019
;; MSG SIZE rcvd: 839

```

xi) Check for the presence of RRSIG records with following command

```
dig A abc.go.tz. @localhost +noadditional +dnssec +multiline
```

Sample output:

```
; <<>> DiG 9.8.4-rpz2+r1005.12-P1 <<>> A abc.go.tz. @localhost
+noadditional +dnssec +multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32902
;; flags: qr aa rd; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL:
5
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;abc.go.tz.      IN A

;; ANSWER SECTION:
abc.go.tz.      86400 IN A 93.184.216.119
abc.go.tz.      86400 IN RRSIG A 7 2 86400 20191227171405 (
                20191127171405 40400 abc.go.tz.

JCoL8L7As1a8CXnx1W62O94eQl6zvVQ3prtNK7BWIW9O
lir/4V+a6c+0tbt4z4lhgmb0sb+qdvqRnlI7CydaSZDb
hlrJA93fHqFqNXw084YD1gWC+M8m3ewbobiZgBUh5W66
1hsVjWZGvvQL+HmobuSvsF8WBMAFgJgYLg0YzBAvwHIk
886be6vbNeAltVPl9I+tjllXkMK5dReMH40ulgKo+Cwb
xNQ+RfHhCQIwKgyvL1JGuHB125rdEQEVnMy26bDcC9R+
qJNYj751CEUZxEEGI9cZkD44oHwDvPgF16hpNZGUdo8P
GtuH4JwP3hDIpNtGTsQrFWYWL5pUuuQRwA== )

;; AUTHORITY SECTION:
abc.go.tz.      86400 IN NS master.abc.go.tz.
abc.go.tz.      86400 IN NS slave.abc.go.tz.
abc.go.tz.      86400 IN RRSIG NS 7 2 86400 20191227171405 (
                20191127171405 40400 abc.go.tz.

hEGzNvKnc3sXkiQKo9/+ylU5WSFWudbUc3PAZvFMjyRA
j7dzcVwM5oArK5eXJ8/77CxL3rfwGvi4LJzPQjw2xvDI
oVKei2GJNYekU38XUwzSMrA9hnkremX/KoT4Wd0K1NPY
giaBgyyGR+PT3jIP95Ud6J0YS3+zg60Zmr9iQPBifH3p
QrvvY3OjXWYL1FKBK9+rJcwzlsSslbmj8ndL1OBKPEX3
psSwneMAE4PqSgbcWtGlzySdmJLKqbI1oB+d3I3bVWRJ
4F6CpIRRCb53pqLvXWQw/NXyVefNTX8CwOb/uanCCMH8
wTYkCS3APl/hu20Y4R5f6xyt8JZx3zkZEQ== )

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Nov 28 00:01:06 2019
;; MSG SIZE rcvd: 1335
```

b) Automatic signing

The other method is using automatic signing. Edit the file `/etc/bind/named.conf.local` to reflect below

```
zone "abc.go.tz" {
    type master;
    file "db.abc.go.tz";
    key-directory "/var/cache/bind";
    auto-dnssec maintain;
    inline-signing yes;
    allow-transfer {2.2.2.2; };
    allow-update { none; };
};
```

Note: The key-directory is the location of the KSK/ZSK keys. The BIND user must have 'read' access to this, so update your permissions accordingly.

auto-dnssec in **section b** above has two options — allow or maintain.

- auto-dnssec allow searches the key directory and signs the zone with the corresponding keys once it receives the command `rndc sign`.
- auto-dnssec maintain does as above but also periodically checks the key directory.

Save this file and restart bind with `service bind9 restart`.

Check for the DNSKEY record using dig on the same server.

```
dig DNSKEY abc.go.tz. @localhost +multiline
```

This marks the end of configuration of the master server.

B. DNSSEC Slave Server Configuration

The slave server only requires DNSSEC to be enabled and the zone file location to be changed to reflect a new one. The following are the steps for configuration.

- i) Edit the main configuration file of BIND with the following command.

```
nano /etc/bind/named.conf.options
```

- ii) Place these lines inside the "options {}" section if they don't exist.

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


iii) Edit the “file” option inside the “zone {}” section.

```
zone "abc.go.tz" IN {  
    type slave;  
    file "abc.go.tz.signed";  
    masters { 1.1.1.1; };  
    allow-notify { 1.1.1.1; };  
};
```

iv) Reload the BIND service.

```
service bind9 reload
```

v) Check if there is a new “.signed” zone file by typing the following syntax in the command prompt.

```
[root@slave ~]# ls -l /var/cache/bind/  
total 16  
-rw-r--r-- 1 bind bind 472 Nov 27 17:25 abc.go.tz.zone  
-rw-r--r-- 1 bind bind 9180 Nov 27 18:29 abc.go.tz.zone.signed
```

5.1.2. Configure DNSKEY records with the registrar

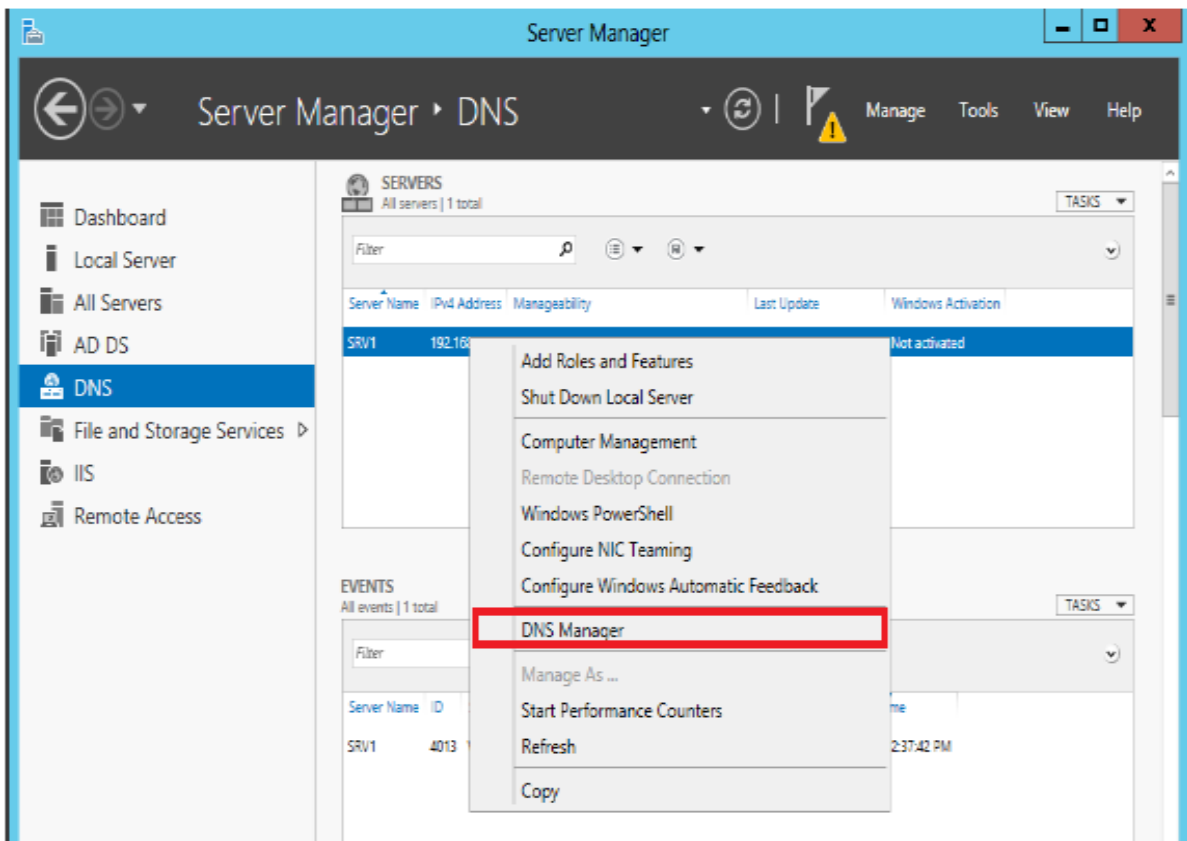
- i) Use below command to get DNSKEY records. Copy the output and share it with your parent zone ie your .TZ registrar. `dig DNSKEY abc.go.tz. @localhost +multiline`
- ii) Once your registrar updated your domain with the above information and the zone is propagated you check if DNSSEC is working fine using any of the following online services.
 - DNSSEC DEBUGGER (<http://dnssec-debugger.verisignlabs.com/>)
 - DNSViz (<http://dnsviz.net/>)

5.2. DNSSEC Deployment for Windows

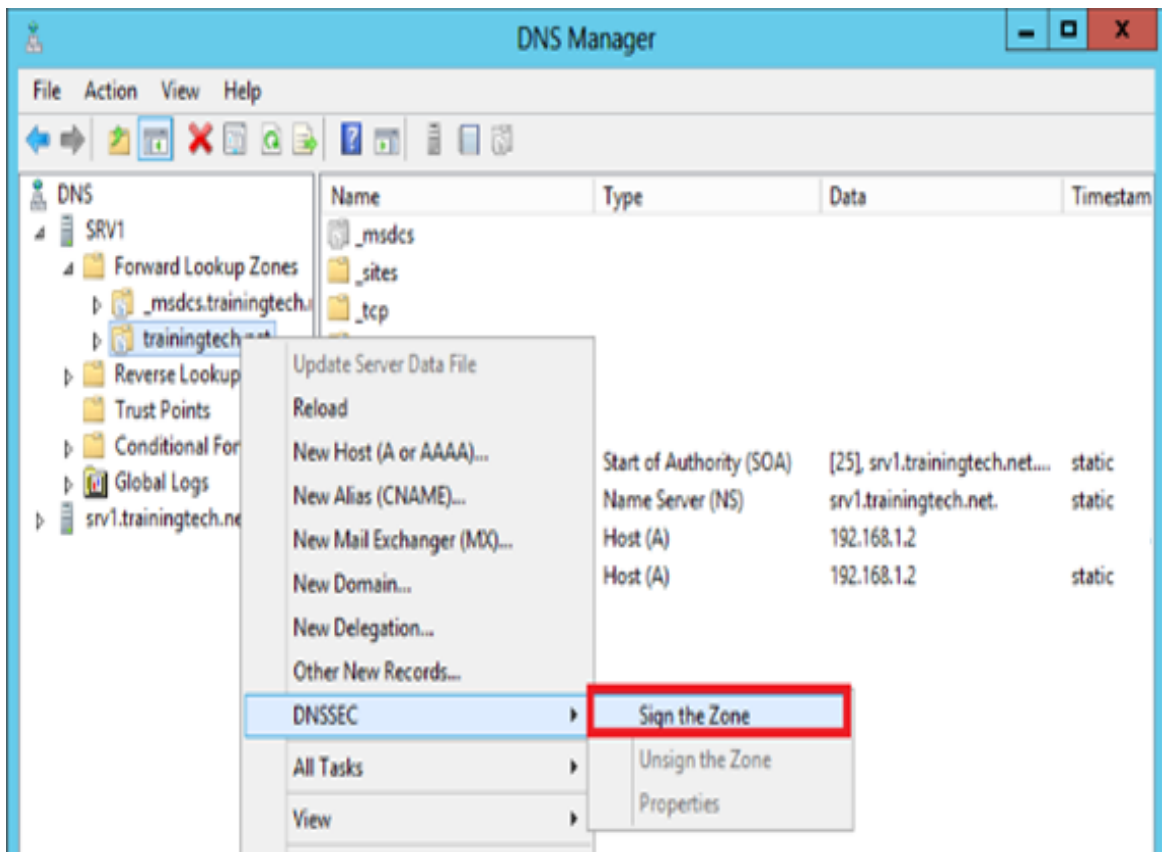
Assumptions: DNS server running on Windows Server 2012 R2 and the domain to deploy DNSSEC is “trainingtech.net”

In Windows Server 2012 R2, DNSSEC is enabled by default. The following are the steps for deploying DNSSEC: -

i) Open Server Manager and then Click DNS Manager.



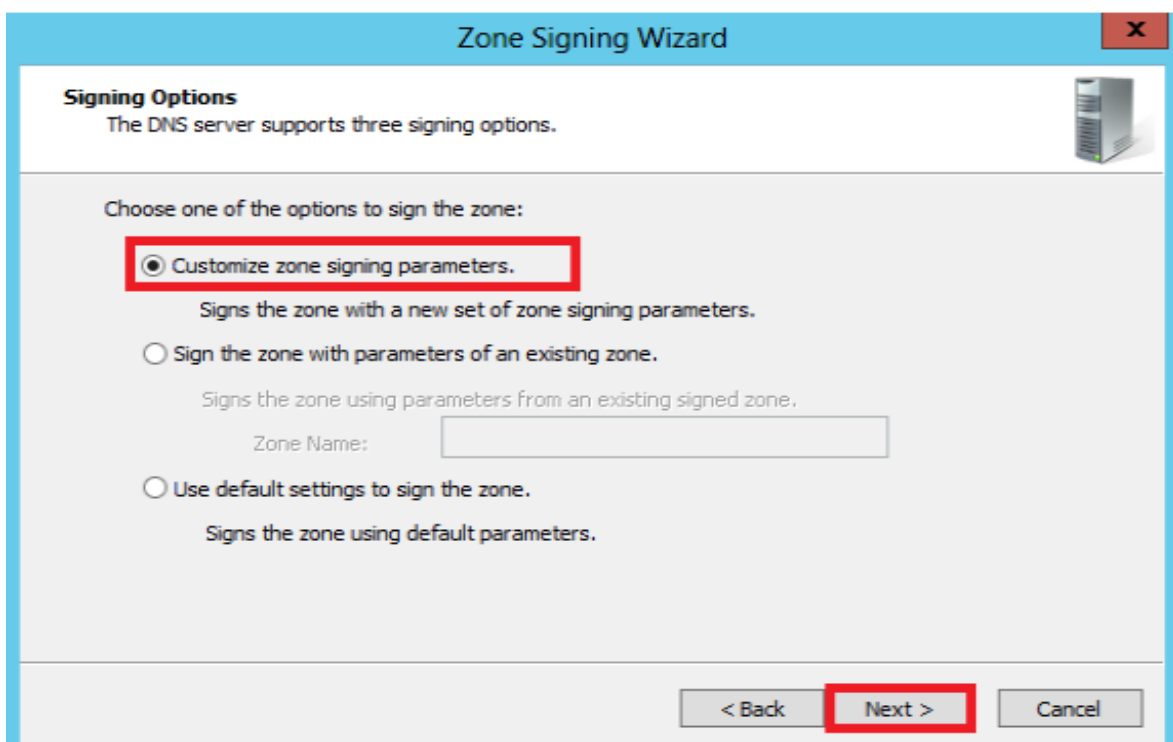
ii) In the DNS Manager console, Select DNSSEC and then select Sign the Zone.



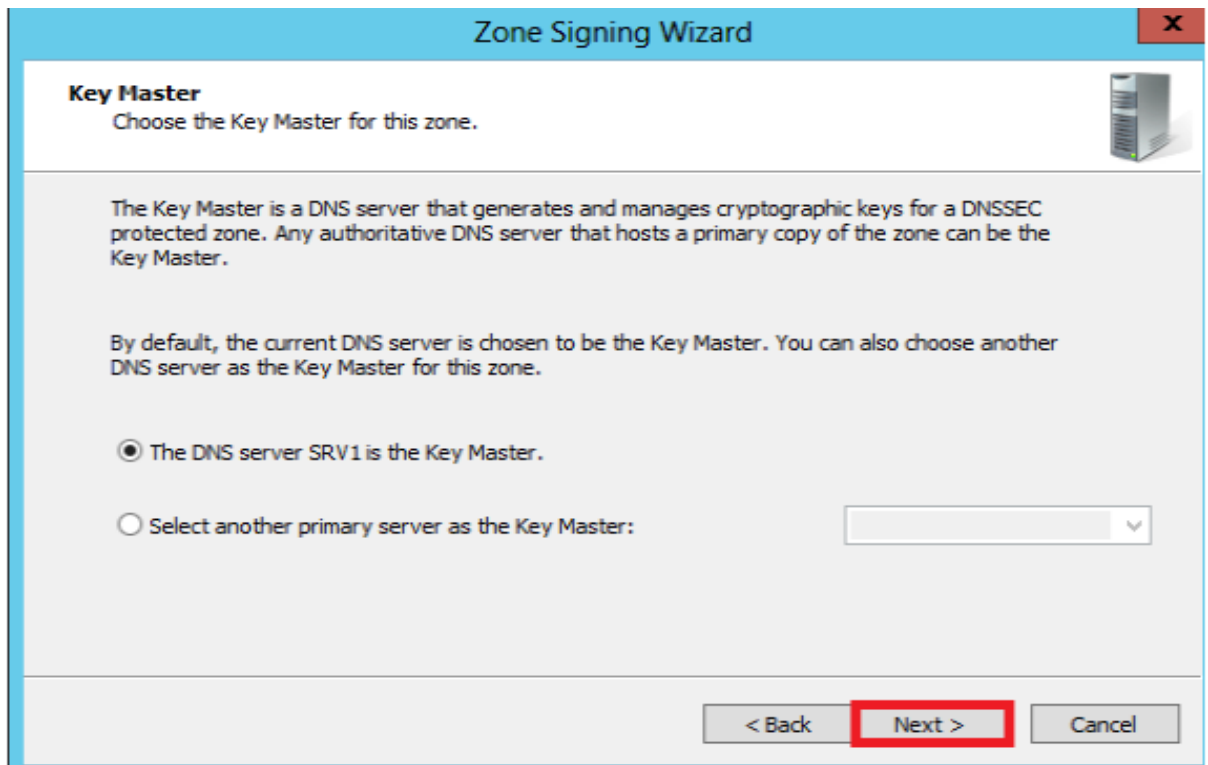
iii) Click Next.



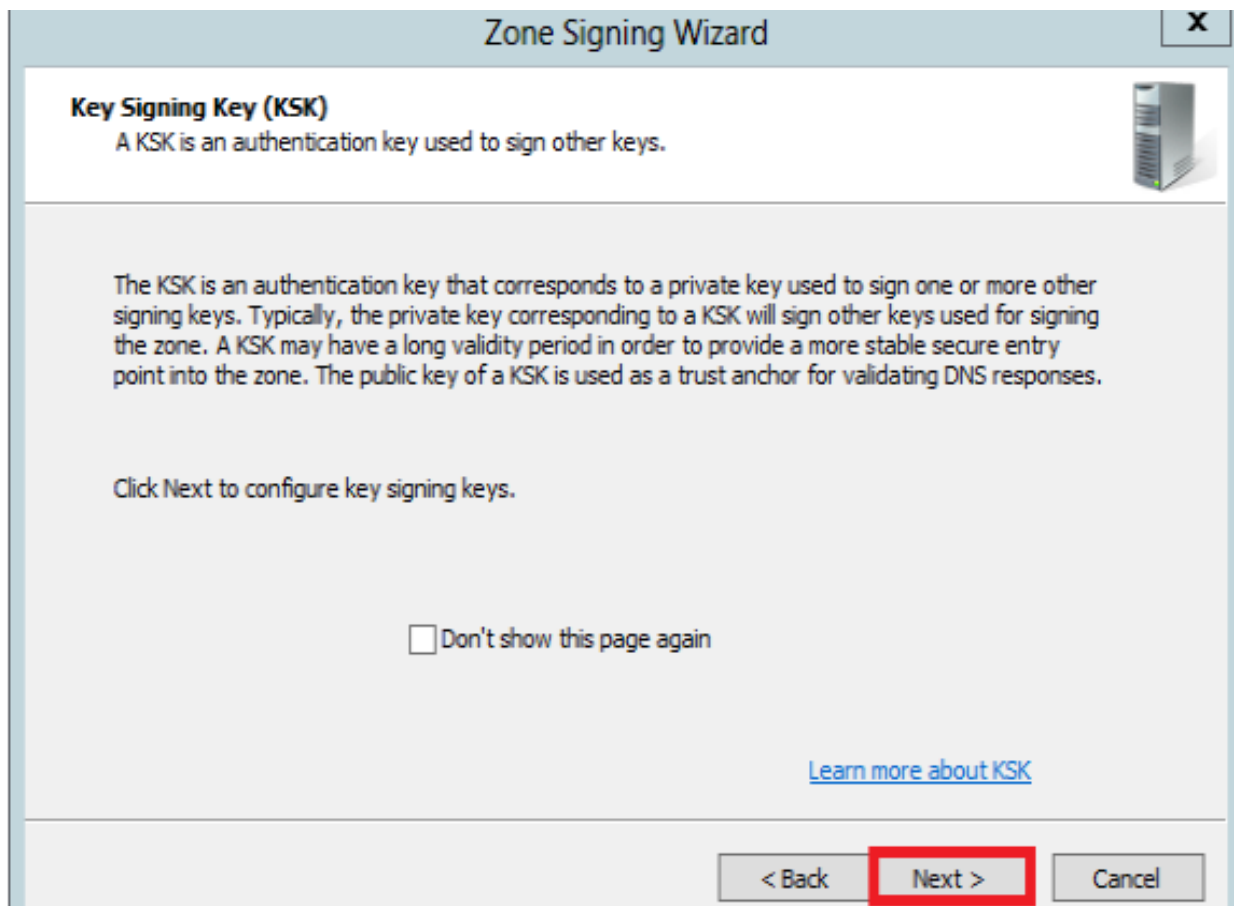
iv) Select Customize Zone Signing Parameters and then Click Next.



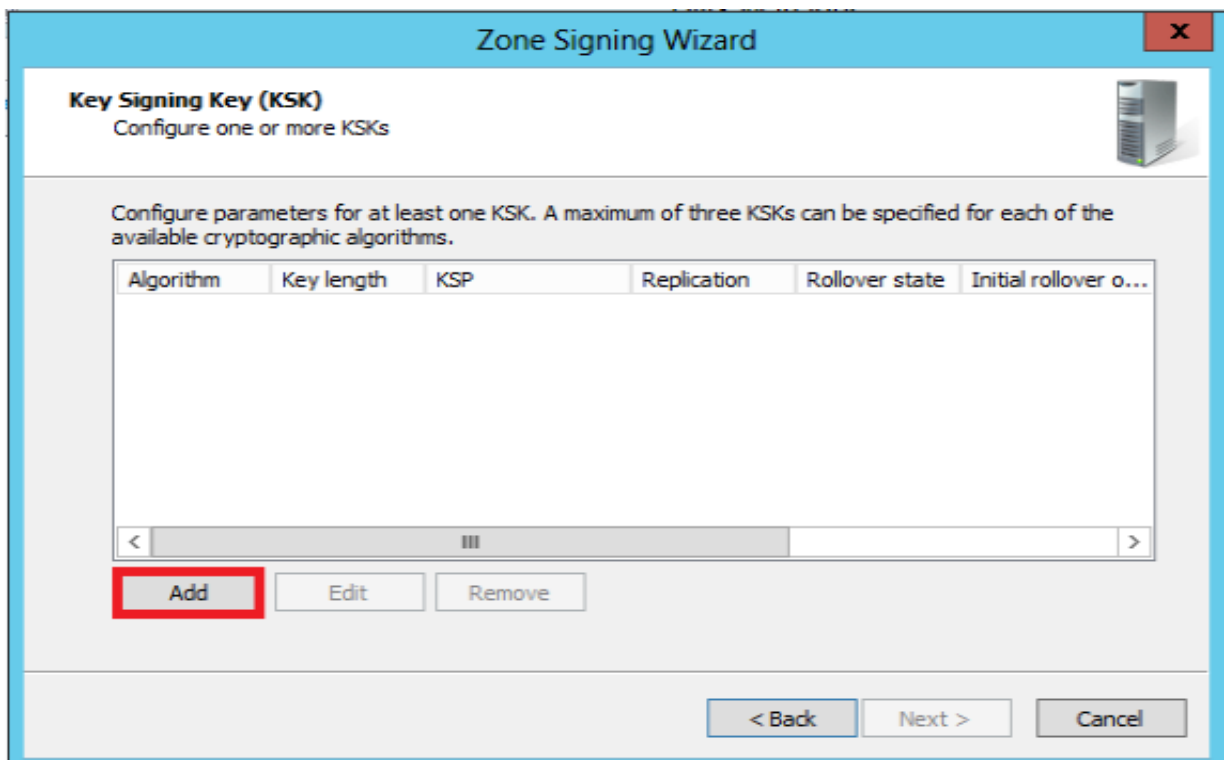
- v) Select one DNS server as the key master for the zone. The key master is responsible for generating new signing keys.



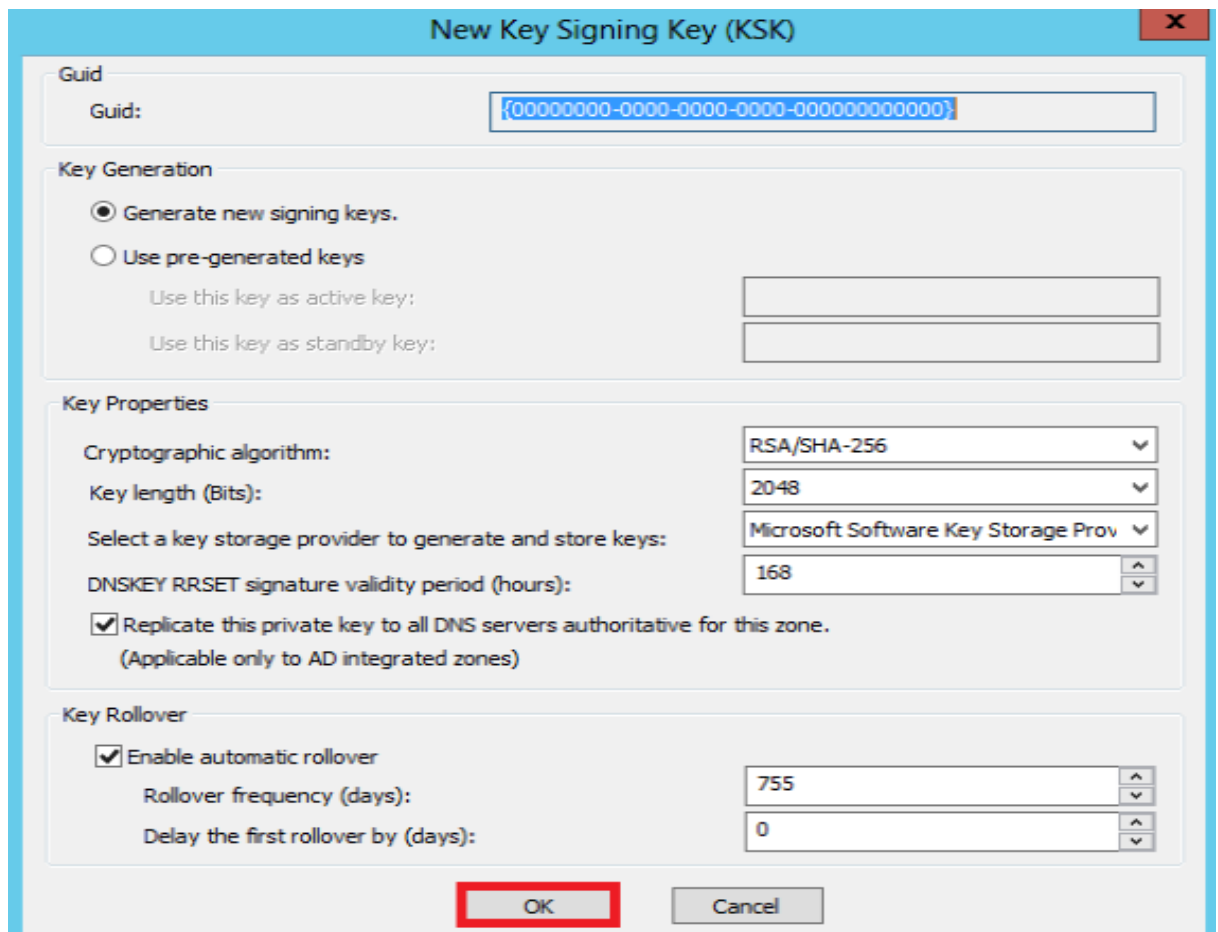
- vi) Click Next.



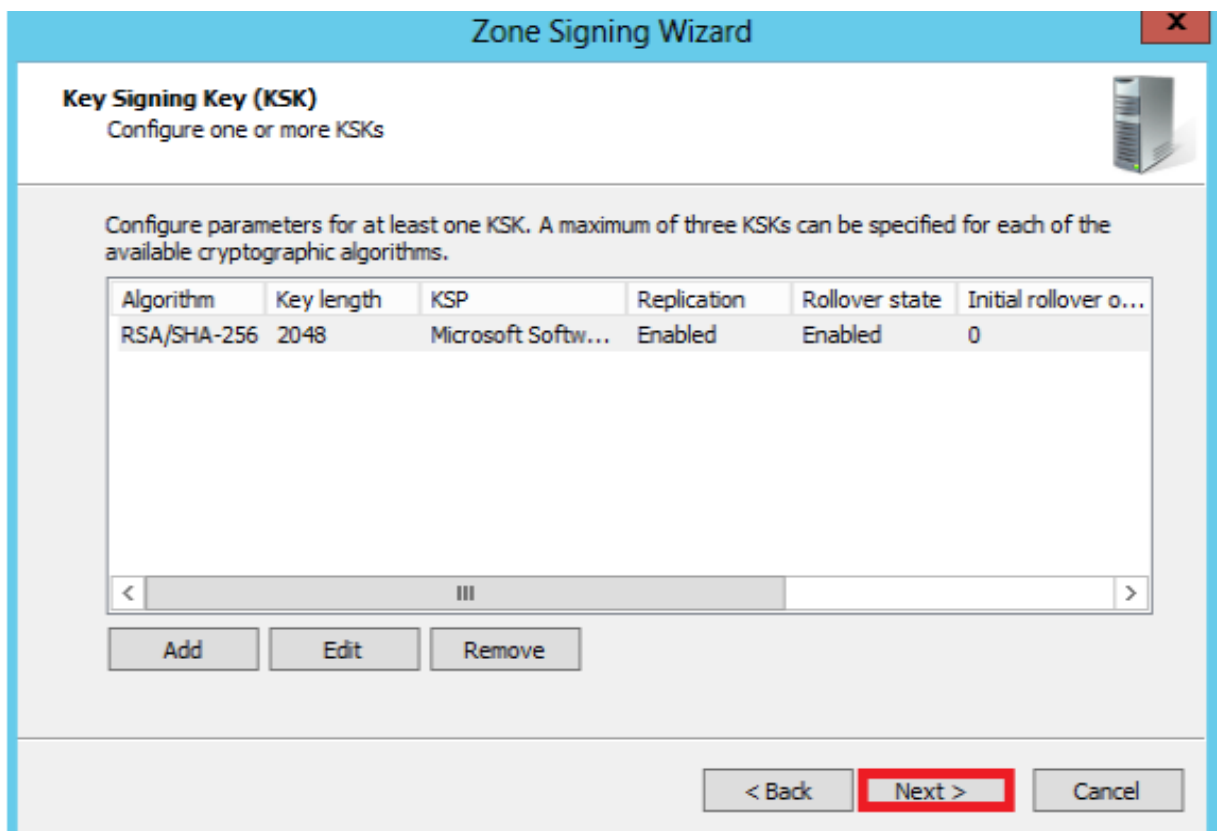
vii) On the key signing key page, Click Add.



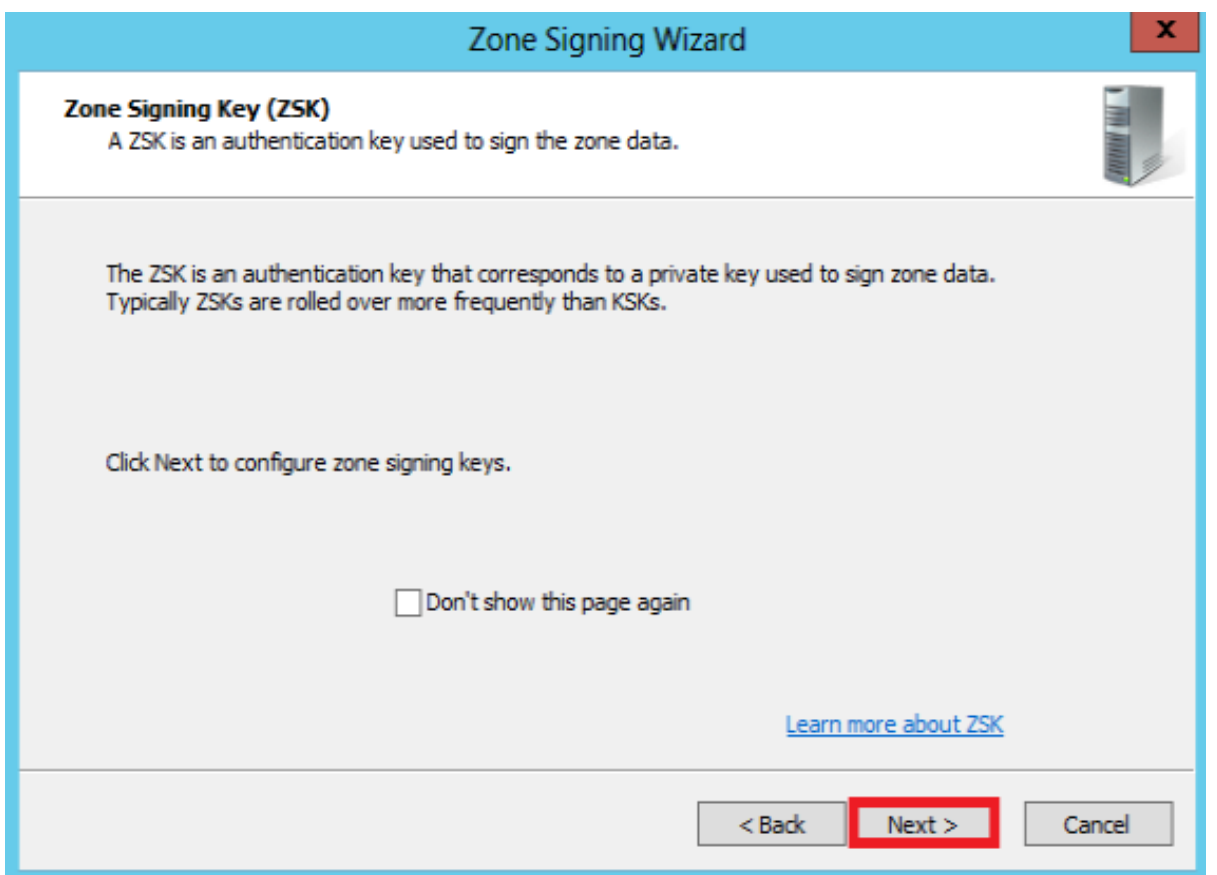
viii) Click Ok.



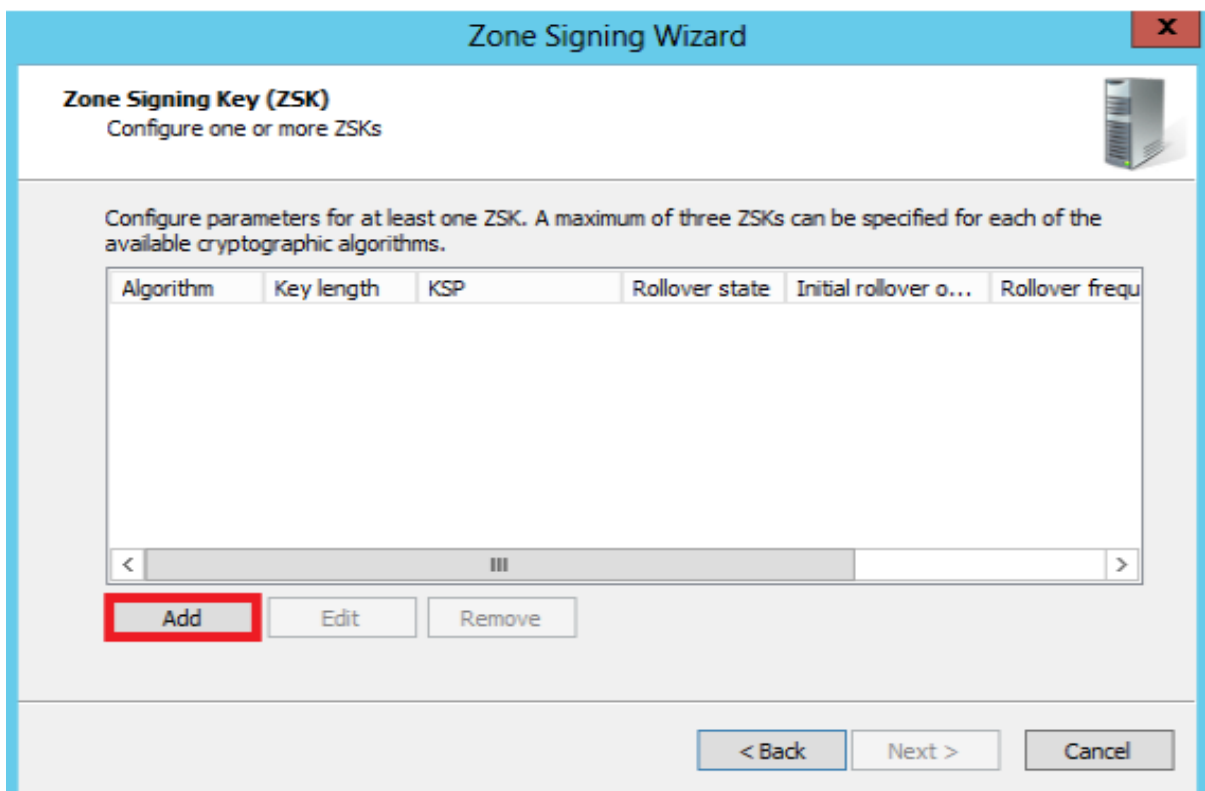
ix) Click Next.



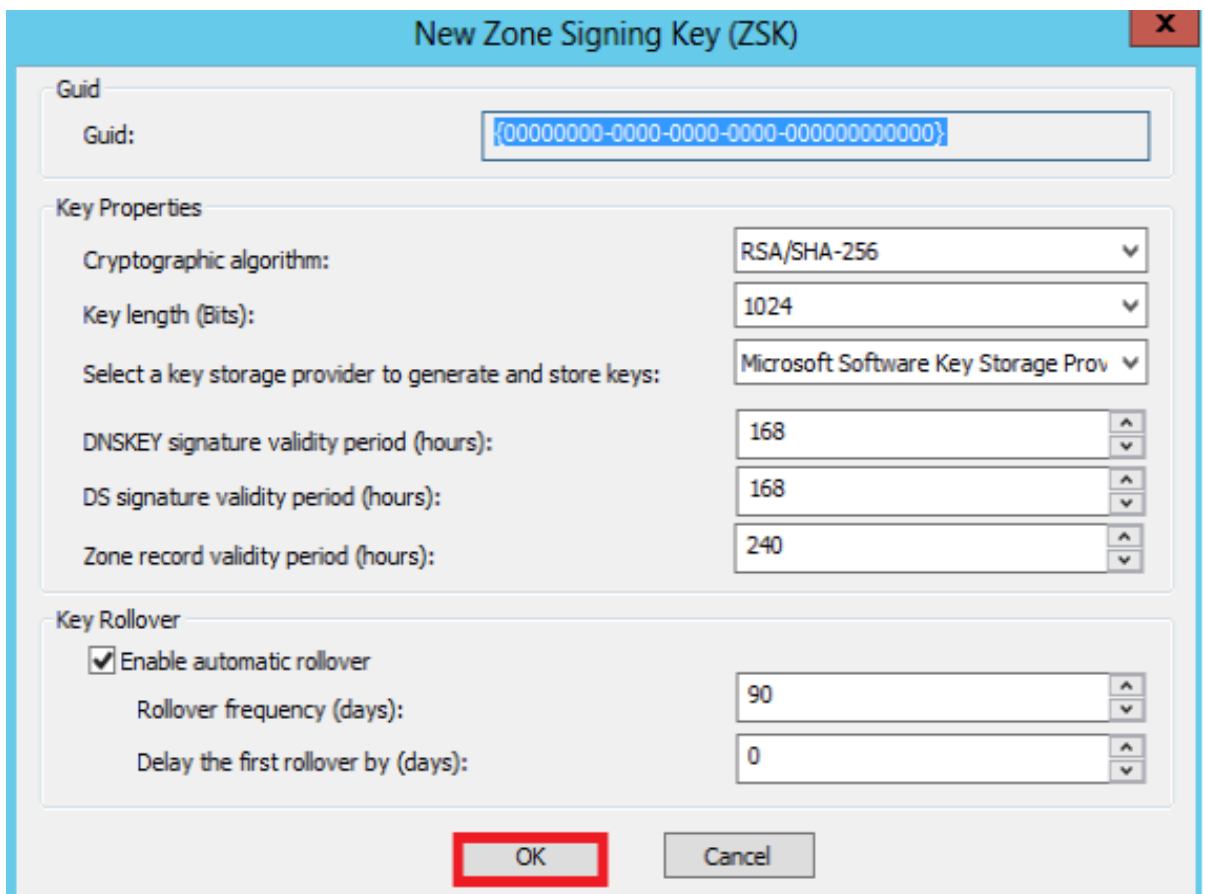
x) On zone signing key, Click Next.



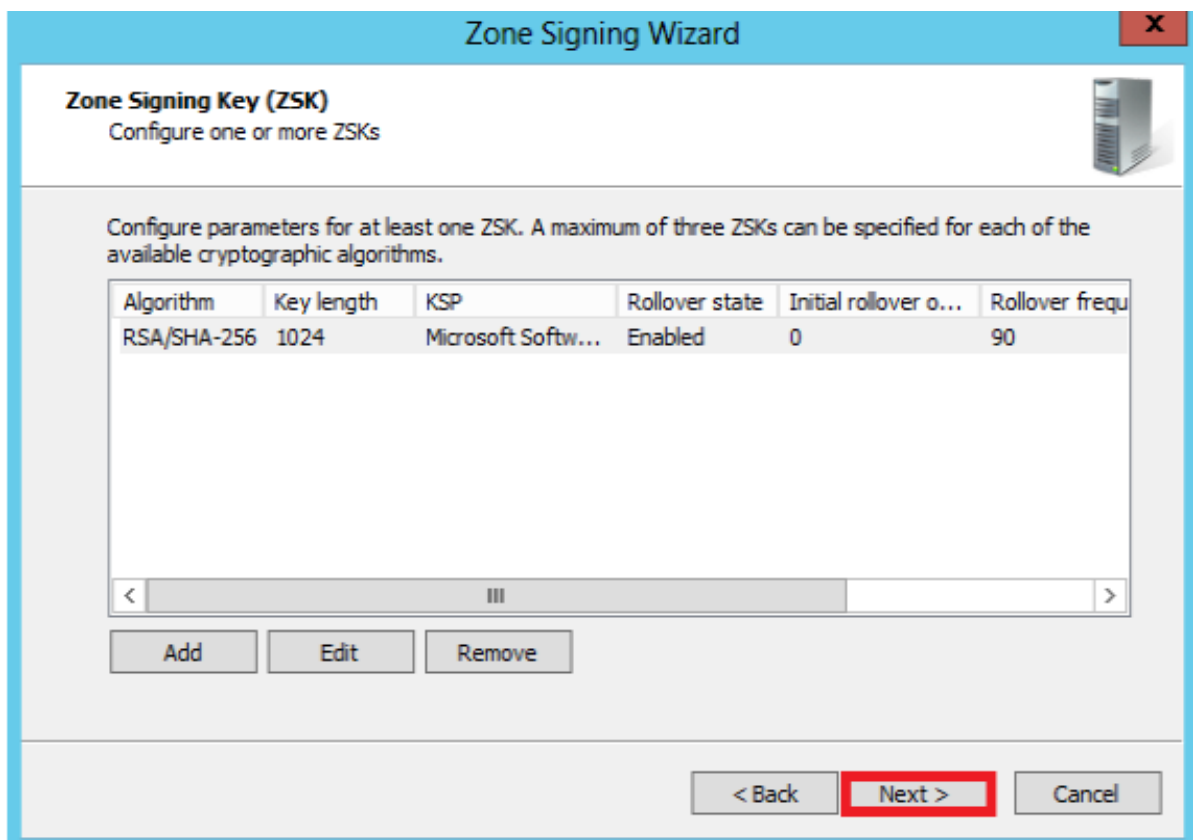
xi) On the Zone Signing Key page, Click Add to configure a ZSK.



xii) Click Ok.



xiii) Click Next.



Zone Signing Wizard

Zone Signing Key (ZSK)
Configure one or more ZSKs

Configure parameters for at least one ZSK. A maximum of three ZSKs can be specified for each of the available cryptographic algorithms.

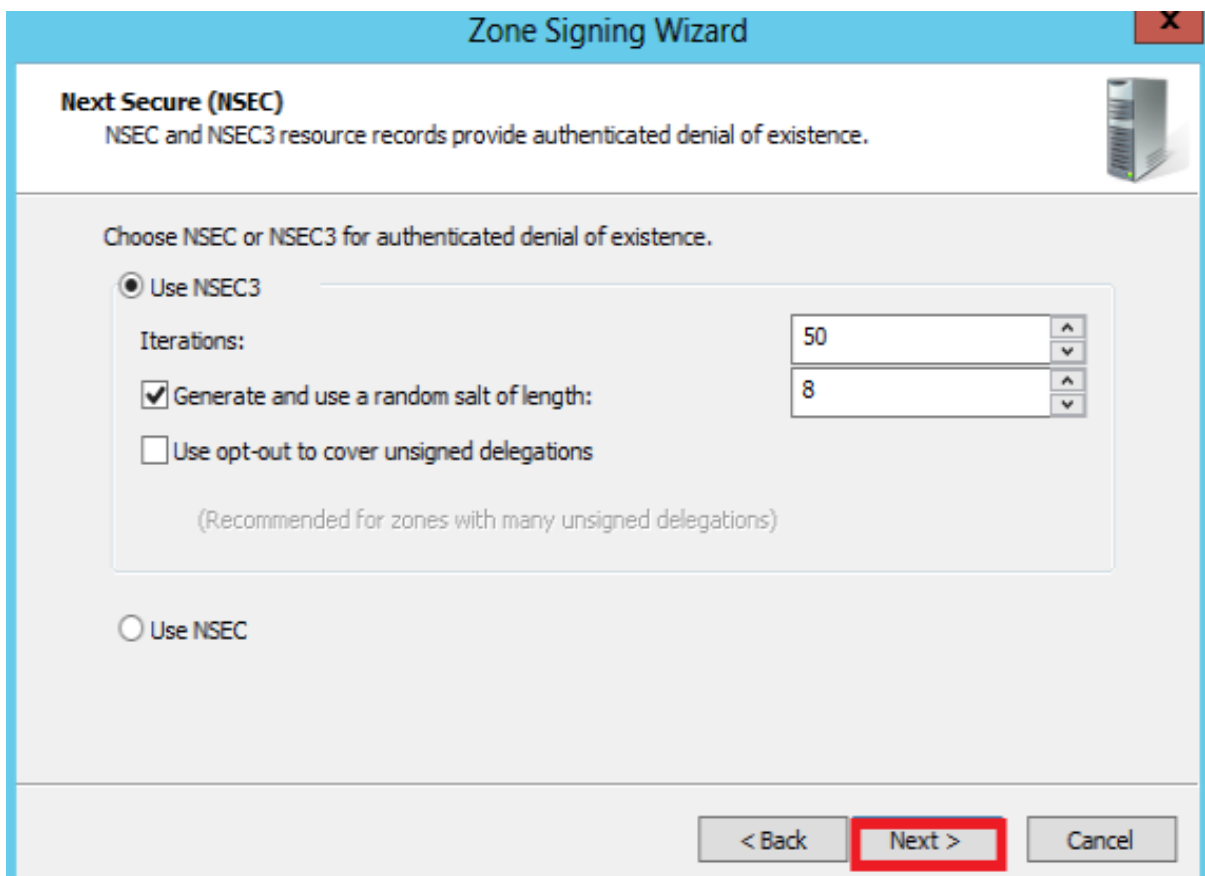
Algorithm	Key length	KSP	Rollover state	Initial rollover o...	Rollover frequ
RSA/SHA-256	1024	Microsoft Softw...	Enabled	0	90

< III >

Add Edit Remove

< Back **Next >** Cancel

xiv) Select NSEC3 resource record rather than the older NSEC resource record for authenticated denial of existence.



Zone Signing Wizard

Next Secure (NSEC)
NSEC and NSEC3 resource records provide authenticated denial of existence.

Choose NSEC or NSEC3 for authenticated denial of existence.

Use NSEC3

Iterations: 50

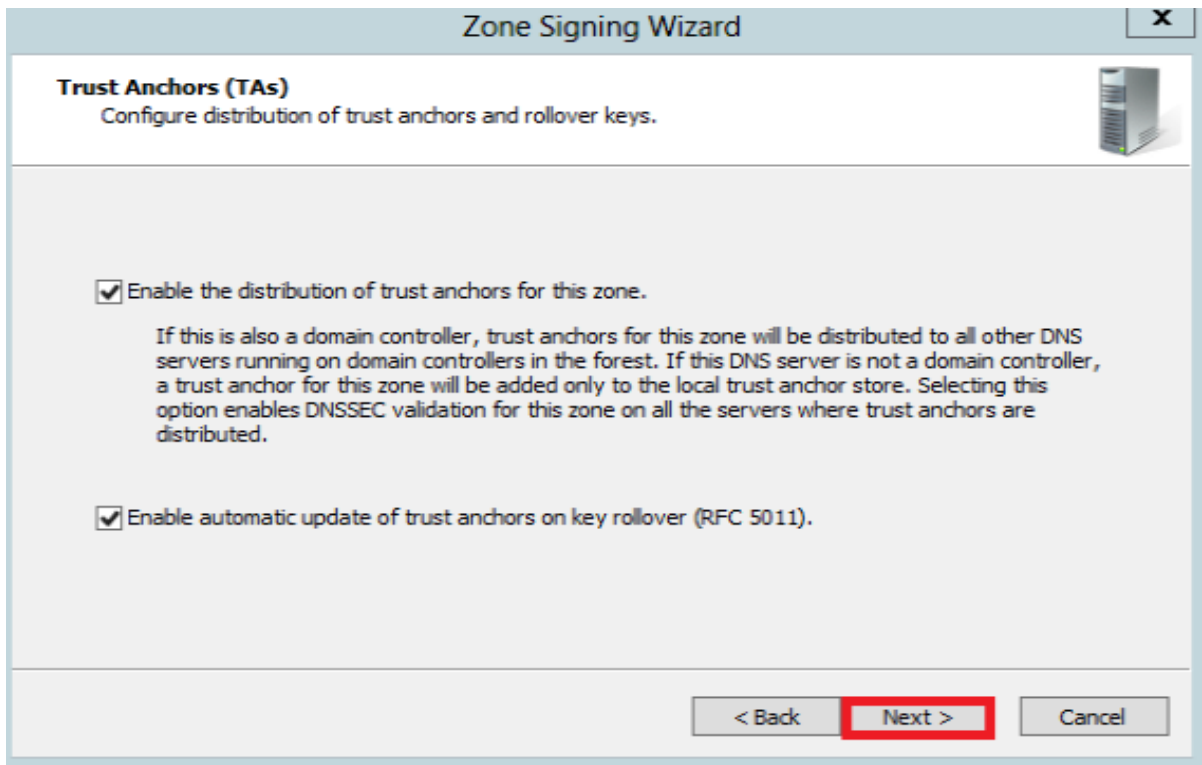
Generate and use a random salt of length: 8

Use opt-out to cover unsigned delegations
(Recommended for zones with many unsigned delegations)

Use NSEC

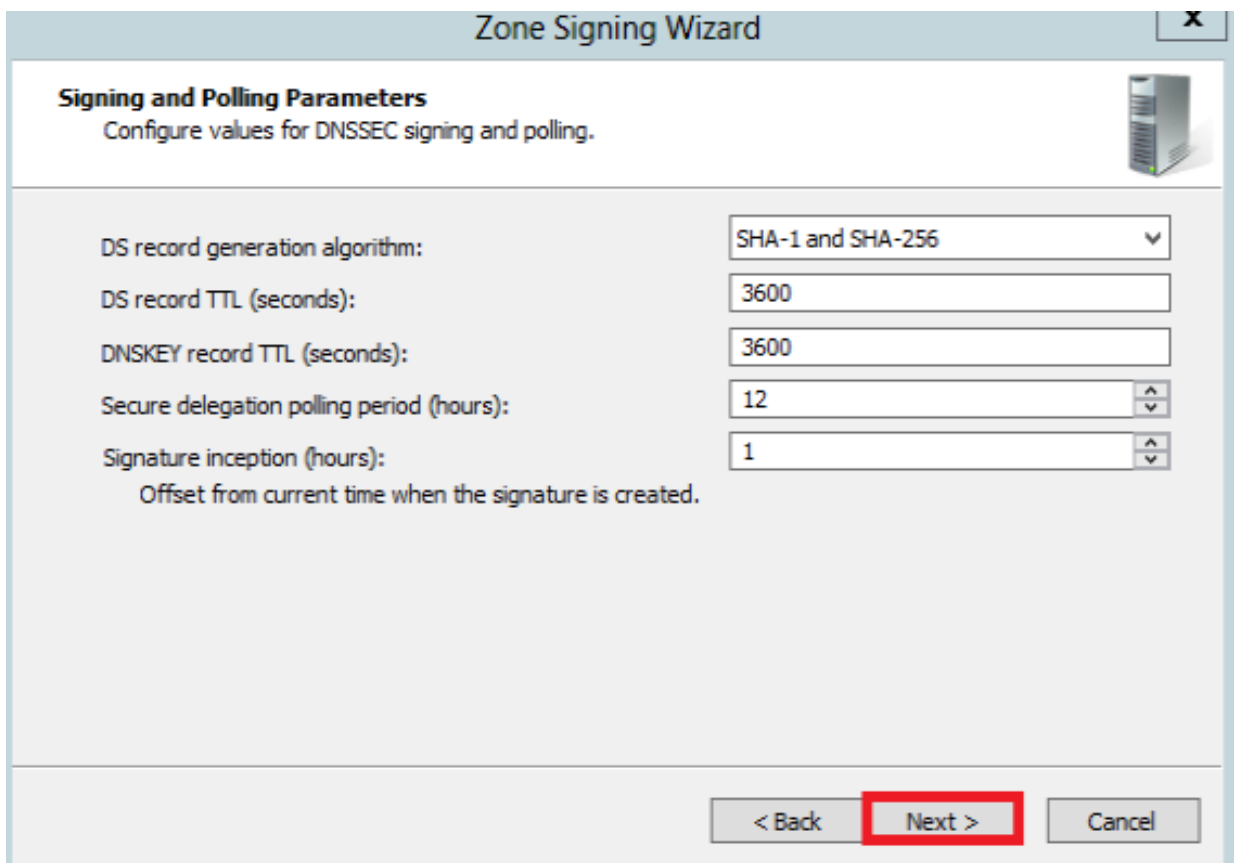
< Back **Next >** Cancel

- xv) By default, trust anchors are updated automatically. You can also enable the distribution of trust anchors for the zone.



The screenshot shows the 'Zone Signing Wizard' window, specifically the 'Trust Anchors (TAs)' step. The title bar reads 'Zone Signing Wizard' with a close button (X) on the right. Below the title bar, the section is titled 'Trust Anchors (TAs)' with the subtitle 'Configure distribution of trust anchors and rollover keys.' and a server icon. The main area contains two checked options: 'Enable the distribution of trust anchors for this zone.' and 'Enable automatic update of trust anchors on key rollover (RFC 5011)'. A detailed explanation is provided for the first option: 'If this is also a domain controller, trust anchors for this zone will be distributed to all other DNS servers running on domain controllers in the forest. If this DNS server is not a domain controller, a trust anchor for this zone will be added only to the local trust anchor store. Selecting this option enables DNSSEC validation for this zone on all the servers where trust anchors are distributed.' At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'. The 'Next >' button is highlighted with a red rectangle.

- xvi) For signing and polling, SHA-1 and SHA-256 are the default algorithms used. Click Next.

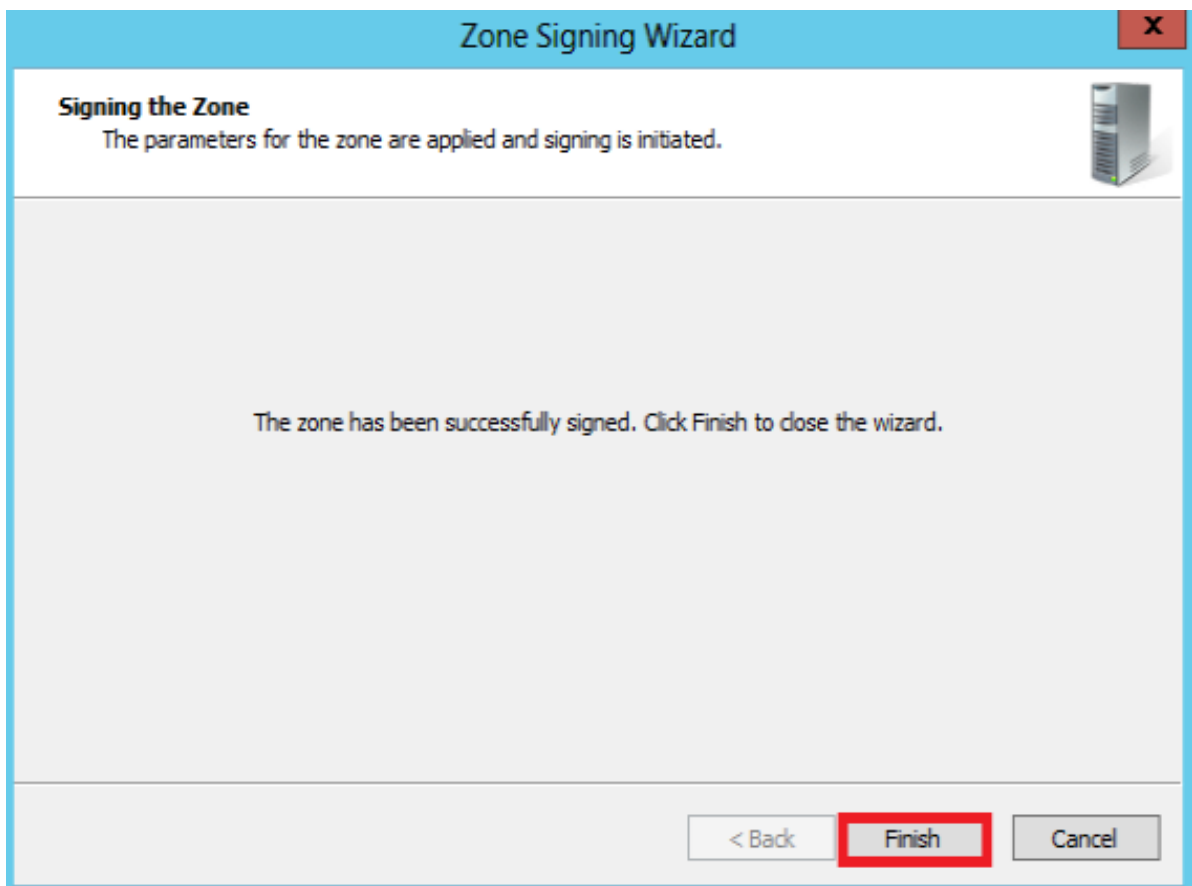


The screenshot shows the 'Zone Signing Wizard' window, specifically the 'Signing and Polling Parameters' step. The title bar reads 'Zone Signing Wizard' with a close button (X) on the right. Below the title bar, the section is titled 'Signing and Polling Parameters' with the subtitle 'Configure values for DNSSEC signing and polling.' and a server icon. The main area contains five configuration fields: 'DS record generation algorithm:' with a dropdown menu set to 'SHA-1 and SHA-256'; 'DS record TTL (seconds):' with a text box containing '3600'; 'DNSKEY record TTL (seconds):' with a text box containing '3600'; 'Secure delegation polling period (hours):' with a spinner box set to '12'; and 'Signature inception (hours):' with a spinner box set to '1'. A note below the last field reads 'Offset from current time when the signature is created.' At the bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'. The 'Next >' button is highlighted with a red rectangle.

xvii) Click Next.



xviii) After the wizard signs the zone, click Finish.



5. CONCLUSION

To guarantee safe and secure functioning of the internet, it is critically important for domain holders to implement DNSSEC on their DNS servers.

6. REFERENCES

- a) RFC 4033 – DNS Security Introduction and Requirements
- b) RFC 4034 – Resource Records for the DNS Security Extensions
- c) RFC 4035 – Protocol Modifications for the DNS Security Extensions
- d) RFC 4470 – Minimally Covering NSEC Records and DNSSEC On-line Signing
- e) RFC 4641 – DNSSEC Operational Practices
- f) RFC 5155 – DNS Security (DNSSEC) Hashed Authenticated Denial of Existence
- g) RFC 6014 – Cryptographic Algorithm Identifier Allocation for DNSSEC
- h) <https://www.digitalocean.com/community/tutorials/how-to-setup-dnssec-on-an-authoritative-bind-dns-server--2>
- i) <https://www.icann.org/resources/pages/dnssec-what-is-it-why-important-2019-03-05-en>
- j) <https://tools.ietf.org/html/rfc7129>
- k) <http://www.trainingtech.net/implementing-dnssec-in-windows-server-2012/>
- l) <https://www.cloudflare.com/learning/dns/what-is-dns/>