Table Of Contents

Table Of Contents 1
How it works? Each name server adds a TSIG record the data section of a dns server-to-server que 2
How do I configure TSIG? 2
Using TSIG - master server configuration 3
Using TSIG - slave server configuration 4
Restrict zone transfers only to those signed with a specific key 4
Verify TSGI 5
Suggested readings: 5

nixCraft: Linux Tips, Hacks, Tutorials, And Ideas In Blog Format
http://www.cyberciti.biz/

Bind Security: Transaction Signatures (TSIG) Configuration

Posted by Vivek Gite <vivek@nixcraft.com>

[1]

Q. How do I configure BIND9 name serves with TSIG (Transaction SIGnature) mechanism to secure server-to-server communication? How do I use secret key transaction authentication for DNS (bind nameservers)?

A. Transaction signatures (TSIG) is a mechanism used to secure DNS messages and to provide secure server-to-server communication (usually between master and slave server, but can be extended for dynamic updates as well). TSIG can protect the following type of transactions between two DNS servers:

- Zone transfer
- Notify
- Dynamic updates
- Recursive query messages etc

TSIG is available for BIND v8.2 and above. TSIG uses shared secrets and a one-way hash function to authenticate DNS messages. TSIG is easy and lightweight for resolvers and named.

How it works?

1. **Each name server adds a TSIG record the data section of a dns server-to-server queries and message.**
2. **The TSIG record signs the DNS message, proving that the message's sender had a cryptographic key shared with the receiver and that the message wasn't modified after it left the sender.**
3. **TSIG uses a one-way hash function to provide authentication and data integrity.**

Our sample setup:

- **Master nameserver: ns1.theos.in - 202.54.1.2**
- **Slave nameserver: ns2.theos.in - 75.55.2.100**
- **BIND configuration is stored in *./etc/bind/* directory.**
- **Zone data is stored in *./etc/bind/named.conf* *file.***

How do I configure TSIG?

Type the following command on master nameserver (ns1.theos.in) to create the shared keys, using the dnssec-keygen program, which creates two files, both containing the key generated.

```
# dnssec-keygen -a HMAC-MD5 -b 128 -n HOST rndc-key
```

Sample output:

```
Krndc-key.+157+64252
```

List all files, enter:

```
# ls -l
```

Output:

```
total 52
-rw-r--r-- 1 root root  237 2009-01-06 12:16 db.0
-rw-r--r-- 1 root root  271 2009-01-06 12:16 db.127
-rw-r--r-- 1 root root  237 2009-01-06 12:16 db.255
-rw-r--r-- 1 root root  353 2009-01-06 12:16 db.empty
-rw-r--r-- 1 root root  256 2009-01-06 12:16 db.local
-rw-r--r-- 1 root root 1506 2009-01-06 12:16 db.root
-rw------- 1 root root   52 2009-01-25 14:13 Krndc-key.+157+64252.key
-rw------- 1 root root   81 2009-01-25 14:13 Krndc-key.+157+64252.private
-rw-r--r-- 1 root bind 1302 2009-01-25 14:13 named.conf
-rw-r--r-- 1 root bind  165 2009-01-06 12:16 named.conf.local
-rw-r--r-- 1 root bind  358 2009-01-25 14:02 named.conf.options
-rw-r----- 1 bind bind   77 2009-01-24 20:37 rndc.key
-rw-r--r-- 1 root root 1317 2009-01-06 12:16 zones.rfc1918
```

Where,

- -a Specify the encryption algorithm.
- -b Specify the key size.
- -n Specify the nametype. A nametype can be a ZONE, HOST, ENTITY, or USER. Usually, you need to use HOST or ZONE such as theos.in

The above dnssec-keygen program created two files as follows. Both .key and .private files are generated for symmetric encryption algorithms such as HMAC-MD5, even though the public and private key are equivalent:

- **Krndc-key.+157+64252.key** - Contains the public key. The .key file contains a DNS KEY record that can be inserted into a zone file.
- **Krndc-key.+157+64252.private** - Contains the private key. The .private file contains algorithm-specific fields.

Using TSIG - master server configuration

Run the following command and note down the Key:

```
# cat Krndc-key.+157+64252.private
```

Sample output:

```
Private-key-format: v1.2
Algorithm: 157 (HMAC_MD5)
Key: 0jnu3SdsMvzzlmTDPYRceA==
Bits: AAA=
```

Open /etc/bind/tsig.key file, enter:

```
# vi /etc/bind/tsig.key
```

Now you need to create tsig.key file on **master server** as follows:

```
key "TRANSFER" {
          algorithm hmac-md5;
          secret "0jnu3SdsMvzzlmTDPYRceA==";
};
# Slave server IP # 1
server 75.55.2.100 {
        keys {
                TRANSFER;
    };
};
################################
# If you have 3rd slave server with IP 64.1.2.3
#server 64.1.2.3 {
#        keys {
#                TRANSFER;
#    };
#};
################################
```

First block is nothing but keys. TSIG keys are configured using the keys substatements. The keys substatements inform a name server to sign queries and zone transfer requests sent to a particular remote name server. In our case the above substatement informs the master server, to sign all requests to the host slave server 75.55.2.100 with the key called TRANSFER. The server statement's keys clause to tell the slave name server to sign all zone transfer requests and queries sent to its master server and vice verse. Save and close the file. Open named.conf file, enter:

```
# vi /etc/bind/named.conf
```

Append the following line:

```
include "/etc/bind/tsig.key";
```

Save and close the file. Restart named:

```
# rndc reload
```

OR

```
# service named restart
```

Using TSIG - slave server configuration

Create /etc/bind/tsig.key on slave server, enter:

```
# vi /etc/bind/tsig.key
```

Append following config:

```
key "TRANSFER" {
 algorithm hmac-md5;
 secret "0jnu3SdsMvzzlmTDPYRceA==";
};
# Master server IP
server 202.54.1.2 {
 keys { TRANSFER; };
};
```

Save and close the file. Append following to named.conf:

```
include "/etc/bind/tsig.key";
```

Restrict zone transfers only to those signed with a specific key

On the master name server, you can restrict zone transfers only to those signed with a specific key such as TRANSFER. open named.conf

```
# vi /etc/bind/named.conf
```

You must restrict zone transfers to those signed with the TRANSFER key as follows:

```
zone "theos.in" {
        type master;
        file "/etc/bind/zones/master.theos.in";
        allow-transfer {  key TRANSFER; };
};
```

Save and close the file. Restart / reload the bind server:

```
# rndc reload
```

OR

```
# service named restart
```

Verify TSGI

Watch your master BIND dns server log file or system log file, enter:

```
# tail -f /var/log/messages
```

OR

```
# tail -f /var/log/syslog
```

OR

```
# grep 'theos.in/IN' /var/log/syslog
```

Sample output:

```
....
....
Jan 26 13:43:11 rose named[9170]: client 75.126.168.152#52204: transfer of 'theos.in/IN': A
Jan 26 13:43:11 rose named[9170]: client 75.126.168.152#52204: transfer of 'theos.in/IN': A
....
..
```

You should able to see similar message on slave server:

```
Jan 26 19:18:33 txvip1 named[17899]: client 208.43.138.52#32806: received notify for zone '
Jan 26 19:18:33 txvip1 named[17899]: zone theos.in/IN: Transfer started.
Jan 26 19:18:33 txvip1 named[17899]: transfer of 'theos.in/IN' from 208.43.138.52#53: conne
Jan 26 19:18:34 txvip1 named[17899]: zone theos.in/IN: transferred serial 2008071011: TSIG
Jan 26 19:18:34 txvip1 named[17899]: transfer of 'theos.in/IN' from 208.43.138.52#53: end o
```

Suggested readings:

- man dnssec-keygen
- BIND 9 Administrator Reference Manual