Secure Time Management

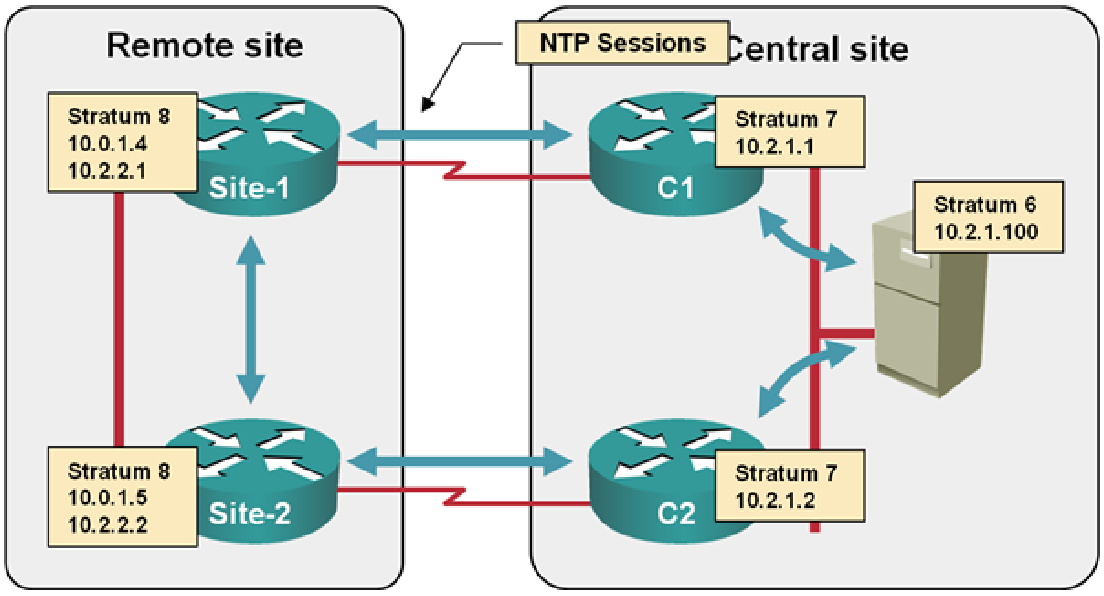
By Ivan Pepelnjak

1. 9. 2008

In the article *It's good to be on time*, I describe how you can use Network Time Protocol (NTP) or its simple variant (SNTP) to synchronize the real-time clock of your network devices (routers and switches) with external time references. As soon as you start using NTP functionality on your routers in a production environment (for example, to expire IPSec certificates), or use network devices as NTP servers to synchronize workstations and servers, NTP becomes a critical part of the networking infrastructure and has to be protected against intruders or impostors (man-in-the-middle attack). For example, if the intruder manages to move the NTP time by a few years in a virtual private network (VPN) using public key infrastructure (PKI), all certificates will expire and IPSec sessions will be dropped, resulting in a very effective denial-of-service (DoS) attack.

In this article, I'll show you how to execute a successful NTP attack on an unprotected network and the safeguards you can put in place to prevent similar attacks. Throughout the article, I'll refer to the sample network in Figure 1.

Figure 1: Diagram of the sample network



The NTP configurations on all routers are extremely simple and contain only NTP server and peer definitions. The sample configuration of the C1 router is displayed in Listing 1. You can get further details on the NTP configuration commands in the IP Corner article *It's good to be on time*.

Listing 1: NTP configuration on C1

```
ntp master

ntp peer 10.2.1.2

ntp server 10.2.1.100
```

Anatomy of an NTP Attack

The sample network has an NTP server that is synchronized with an Internet-based NTP server; its stratum is thus fairly high, as you can see from the **show ntp associations** printout taken from one of the core routers (see Listing 2).

Listing 2: NTP status on C1

```
C1#show ntp associations

      address    ref clock     st  when   poll  reach delay   offset     disp

 ~10.2.1.2      10.2.1.100     7    47     512   357     4.1    15.71      5.4

*~10.2.1.100    127.127.7.1    6    52      64   337     3.9   520.08      3.1

+~127.127.7.1   127.127.7.1    7    48      64   377     0.0     0.00      0.0
```

```
 * master (synced), # master (unsynced), + selected, -
 candidate, ~ configured
```

With the willingness of Cisco IOS to accept anyone as an NTP peer, an intruder can exploit the stratum of the primary NTP server in the network, pose as a stratum 1 (primary) server and inject random time into the network.

Inserting bogus NTP time in the network is as easy as connecting an IP host anywhere in the network, configuring a primary NTP server on it and configuring one of the top-level routers (C1, in this example) as its NTP peer. In our scenario, the intruder will use a Cisco router to simplify the configuration; the relevant configuration commands of the intruder router are shown in Listing 3.

Listing 3: Intruder's NTP configuration

```
ntp master 1

ntp peer 10.2.1.1
```

As soon as the intruder starts sending NTP packets to C1, C1 reports a new NTP peer. The details of the NTP association show the peer to be *dynamic* and *insane* (as its time is years off the current time), as shown in the **show ntp associations detail** printout in Listing 4.

Listing 4: Details of NTP association between C1 and the intruder

```
10.2.1.42 dynamic, insane, invalid, stratum 1

ref ID .LOCL., time F59C845C.3541BD8F (00:03:40.208 UTC Wed Jul 31 2030)

our mode passive, peer mode active, our poll intvl 64, peer poll intvl 64

root delay 0.00 msec, root disp 1875.03, reach 16, sync dist 9758.469

delay 10.85 msec, offset 695979164670.6102 msec, dispersion 7878.02

precision 2**24, version 3

org time F59C848A.3541A5E6 (00:04:26.208 UTC Wed Jul 31 2030)

rcv time CC20B7ED.8F84CD88 (16:31:41.560 UTC Thu Jul 10 2008)

xmt time CC20B7F5.7CC65398 (16:31:49.487 UTC Thu Jul 10 2008)

filtdelay =    51.09   10.85    0.00    0.00    0.00    0.00    0.00

filtoffset = 6959791 6959791    0.00    0.00    0.00    0.00    0.00

filterror =     0.87    1.85 16000.0 16000.0 16000.0 16000.0 16000.0
```

However, after a while, once C1 collects enough statistical data to start trusting the new peer, it selects the intruder as the master server, changes its own stratum to 2 (since it has direct contact with a stratum 1 server) and eventually changes its time (see Listing 5).

Listing 5: C1 selects intruder as its preferred NTP server

```
C1#show ntp associations detail | begin 10.2.1.42

10.2.1.42 dynamic, our_master, sane, valid, stratum 1

ref ID .LOCL., time F59C89DC.353B3CDF (00:27:08.207 UTC Wed Jul 31 2030)

our mode passive, peer mode active, our poll intvl 64, peer poll intvl 64

root delay 0.00 msec, root disp 0.03, reach 376, sync dist 23.911

delay -25.39 msec, offset 695979164653.2522 msec, dispersion 10.86

precision 2**24, version 3

org time F59C89F1.353B1BA4 (00:27:29.207 UTC Wed Jul 31 2030)

rcv time CC20BD54.8FDE044E (16:54:44.561 UTC Thu Jul 10 2008)

xmt time CC20BD75.8382F5EF (16:55:17.513 UTC Thu Jul 10 2008)

filtdelay =    36.36    27.92    20.14    19.97    -0.03     4.00   -25.39

filtoffset = 6959791 6959791 6959791 6959791 6959791 6959791 6959791

filterror =     0.49     1.46     2.44     3.42     4.39     5.37     6.35

C1#

%SYS-6-
CLOCKUPDATE: System clock has been updated from 16:56:52 UTC Thu Jul 10 2008
to 00:29:37 UTC Wed Jul 31 2030, configured from NTP by 10.2.1.42.
```

The change is then slowly propagated throughout the network, resulting in a situation in which an intruder was able to shift the network's perception of current date/time by 22 years, using very simple tools.

Note: NTP reacts very slowly to configuration changes or changes in peer status (time shift, stratum change), making the NTP network troubleshooting a particularly long and boring exercise. Thus, it's best that you properly design and implement NTP.

Access List Protection

Similarly to other network services (for example, Telnet, SSH or HTTP access to the router), you can protect the NTP service with an access control list (ACL). Since NTP is an ancient IOS functionality, you cannot use named access lists. It's best if you stay with standard (non-extended) access lists, as you're only matching the IP address of the sender.

The ACL protection of the NTP service is configured with the **ntp access-group peer | serve | serve-only | query-only** *acl* global configuration command. Each time an incoming NTP packet is received, it's processed by following these steps:

- If the source IP address matches the access list specified in the **ntp access-group peer** configuration command, the router will respond to the packet and will accept the time from the NTP peer if the peer's stratum is lower than the router's own stratum.
  *Warning: If you want a router to become an authoritative NTP server, the **ntp access-group peer** access list should permit the IP address 127.127.7.1 (the internal NTP server created with the **ntp master** configuration command); otherwise, the router will never synchronize to its internal clock.*
- If the source IP address matches the access list specified in the **ntp access-group serve** configuration command, the router will respond to the packet, but will never synchronize to the peer.
- Peers matched by the **ntp access-group serve-only** access list can send only time synchronization packets, not the NTP control packets.
- Similarly, time synchronization packets received from peers matched with the **ntp access-group query-only** access list will be dropped. Only NTP control packets are accepted from peers matched by the **query-only** access list.

If you don't configure **ntp access-group** functionality, all NTP packets are accepted and the router will synchronize with any NTP host claiming to be its peer (the default functionality exploited in the NTP attack described in the previous section). As soon as you configure the first **ntp access-group**, all packets not matched by any of the NTP access groups are dropped. For example, if you configure only the **ntp access-group peer** *acl* (to protect the router), no other NTP client can synchronize with this router; you also have to configure the **ntp access-group serve|serve-only** *acl*.

*Warning: The **ntp access-group** configuration command using a nonexistent access list number matches all packets, potentially resulting in an NTP vulnerability.*

ACL Protection in the Sample Network

The core routers running NTP in the sample network will be protected with two NTP access lists (see Listing 6):

- **access-list 90** used with the **ntp access-group peer** command will match all NTP peers and servers (the other core router and the central NTP server).
- **access-list 91** used with the **ntp access-group serve** command will match loopback interfaces of all routers in the network (the 10.0.1.0/24 address range).

Listing 6: NTP configuration with access lists on C1

```
access-list 90 remark NTP peers & servers

access-list 90 permit 10.2.1.2

access-list 90 permit 10.2.1.100

!

access-list 91 remark NTP clients

access-list 91 permit 10.0.1.0 0.0.0.255

!

ntp logging

ntp access-group peer 90

ntp access-group serve 91

ntp master

ntp peer 10.2.1.2

ntp server 10.2.1.100
```

Similarly, the remote site routers will use two access lists: the ACL used with the **ntp access-group peer** command will list the nearest central router and the LAN NTP peer; the ACL used with the **ntp access-group serve** command will allow all LAN hosts to access the NTP services on the router (see Listing 7).

Listing 7: NTP configuration on a remote site router

```
interface FastEthernet 0/0
 ip address 10.2.2.1 255.255.255.0
!
access-list 90 remark NTP peers & servers
access-list 90 permit 10.0.1.1
access-list 90 permit 10.2.2.2
!
access-list 91 remark NTP clients
access-list 91 permit 10.2.2.0 0.0.0.255
!
ntp logging
ntp access-group peer 90
ntp access-group serve 91
ntp server 10.0.1.1 source Loopback0
ntp peer 10.2.2.2
```

Note: The central routers respond to the NTP packets only if they originated from the loopback interfaces of the remote routers. To specify the NTP source IP address, use the ***ntp server address source interface*** *global configuration command.*

NTP Authentication

NTP authentication allows NTP hosts to authenticate their *servers* and *peers*. It's not used to authenticate NTP clients; NTP servers don't care about the authenticity of their clients, as they never accept any information from them. The implementation of NTP authentication in Cisco IOS is a widely misunderstood topic. It's so poorly explained that many people believe it's been broken in Cisco IOS for years. Fortunately, that's not true; you just need to understand how it works.

Technology Basics

NTP uses very simple authentication using DES-generated checksums. The checksums are computed from the data within the NTP packets and a secret shared between NTP peers (see Appendix C of RFC 1305 for more details). The secret used in calculating the checksum is indicated with a *key identifier* field in the NTP packet. Although each NTP host can have numerous keys, the *key identifier* numbering scheme has to match between the NTP peers.

Configuring NTP Key Infrastructure

The first step in NTP authentication deployment is design and deployment of the NTP keys (shared secrets). In a simple NTP design you should use the same key on all routers, but if you've built a hierarchical NTP network (similar to our sample network), you might want to use multiple NTP keys, one for each level of hierarchy.

Design Tip: Multiple NTP keys are recommended if different departments or groups of engineers manage different levels in the network hierarchy. For example, you don't want to store the core NTP key in easily accessible remote site routers.

In the sample network, we'll use three NTP keys as described in Table 1.

Table 1: NTP keys used in the sample network

Key ID	Value	Usage
222	CoreKey	Peering between the core routers
333	PrimaryKey	Time exchange between the core routers and the primary NTP server
432	RemoteKey	Remote sites

The NTP keys are configured with the **ntp authentication-key** ***number*** **md5** ***password*** global configuration command. In the sample network, all three keys have to be configured on the core routers (see Listing 8), but only a single key has to be configured on the primary NTP server or the remote sites.

Listing 8: NTP keys on the C1 router

```
ntp authentication-key 222 md5 CoreKey

ntp authentication-key 333 md5 PrimaryKey

ntp authentication-key 432 md5 RemoteKey
```

*Note: The passwords specified in the **ntp authentication-key** command are encrypted with type-7 (reversible) encryption even when **no service password-encryption** is configured.*

Authenticating NTP Packets

If you want to sign packets sent from a router to its NTP servers or peers, you have to specify the key identifier to use for each server or peer, using the **ntp peer|server *address* key *number*** global configuration command. When receiving a packet from a peer, the packet is accepted as long as the key identifier in the packet specifies a known password and the checksums match. IOS does not enforce the match between the inbound key identifier and the value configured with the **ntp peer key** configuration command.

Warning: If you don't specify the key identifier for NTP servers or peers, the packets sent to them will not be authenticated.

You don't have to specify the key identifiers used with the clients; IOS automatically selects the correct key. When responding to a client's packet, IOS will use the authentication key specified in the client's packet, assuming that the key identifier is configured on the router and the checksums match. If the key identifier is unknown or the checksums do not match (mismatch in configured passwords), the reply packet will not be authenticated.

Warning: IOS will not report any error if the incoming NTP packet has an invalid authentication key or if there's a mismatch in the packet's checksum and computed checksum.

Should this be "…the packet's checksum and the computed checksum"? In other words, two checksums are being compared to each other? Or perhaps the apostrophe in "packet's" is just a typo, not indicating a possessive?

In the sample network, the NTP authentication key identifiers will be configured on all servers and peers. Listing 9 shows the relevant configuration on C1, and Listing 10 shows the equivalent configuration on a remote router.

Listing 9: NTP key usage on C1

```
ntp server 10.2.1.100 key 333

ntp peer 10.2.1.1 key 222

Listing 10

NTP key usage on Site-1

ntp authentication-key 432 md5 RemoteKey

ntp server 10.2.1.1 key 432

ntp server 10.2.1.1 source loopback 0

ntp peer 10.2.2.2 key 432
```

The authentication keys used in each NTP packet can be inspected with the **debug ntp authentication** printouts. This debugging command reports only the authentication key for each packet. It should be used in combination with the **debug ntp packet peer** ***address*** command to allow you to see the whole picture. For example, when *Site-1* sends an NTP packet to *C1*, the reply has the same authentication key, proving that the key identifier is configured on both routers and that both routers use the same password (see Listing 11)

Listing 11: NTP debugging on Site-1

```
Site-1#show debug

NTP:

  NTP packets debugging is on for peer address 10.2.1.1

  NTP authentication debugging is on

Site-1#

03:48:28: NTP: xmit packet to 10.2.1.1:

03:48:28:  leap 3, mode 3, version 3, stratum 0, ppoll 64

03:48:28:  rtdel 1CBC (112.244), rtdsp 11DC90 (17861.572), refid 0A020101 (1
0.2.1.1)

03:48:28:  ref CC21C26B.467A2AF8 (11:28:43.275 UTC Fri Jul 11 2008)

03:48:28:  org 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)
```

```
03:48:28:  rec 00000000.00000000 (00:00:00.000 UTC Mon Jan 1 1900)

03:48:28:  xmt CC21C2AA.3DD82F4A (11:29:46.241 UTC Fri Jul 11 2008)

03:48:28:  Authentication key 432

03:48:28: NTP: rcv packet from 10.2.1.1 to 10.0.1.4 on Loopback0:

03:48:28:  leap 0, mode 4, version 3, stratum 7, ppoll 64

03:48:28:  rtdel 0E72 (56.427), rtdsp 0240 (8.789), refid 0A020164 (10.2.1.1
00)

03:48:28:  ref CC21C250.DF8D07C7 (11:28:16.873 UTC Fri Jul 11 2008)

03:48:28:  org CC21C2AA.3DD82F4A (11:29:46.241 UTC Fri Jul 11 2008)

03:48:28:  rec CC21C2AA.445C0A79 (11:29:46.267 UTC Fri Jul 11 2008)

03:48:28:  xmt CC21C2AA.446A057B (11:29:46.267 UTC Fri Jul 11 2008)

03:48:28:  inp CC21C2AA.4C34B47B (11:29:46.297 UTC Fri Jul 11 2008)

03:48:28:  Authentication key 432
```

Enabling NTP Authentication

After the NTP keys have been configured and associated with peers or servers, you have to enable NTP time-source authentication with the **ntp authenticate** global configuration command. You also have to specify the *trusted keys*: time information will be accepted only from sources sending packets successfully authenticated with one of the trusted keys.

*Warning: Time information will also be accepted from an unauthenticated peer or server if the peer/server has no key identifier associated with it via the **ntp peer|server key** global configuration command.*

The trusted keys are configured with the **ntp trusted-key *identifier*** global configuration command. You can specify multiple trusted keys. For example, the core routers in the sample network have two trusted keys: the key used by the primary NTP server, and the key used for the core router peerings (see Listing 12)

Listing 12: NTP authentication configured on C1

```
ntp authenticate

ntp trusted-key 222

ntp trusted-key 333
```

Verifying NTP Authentication

Verification of proper operation of NTP authentication in Cisco IOS is complex, as there's no mechanism that would report an error if there's a mismatch in the authentication key or the shared password. You can use the **debug ntp authentication** or **debug ntp validity** command to display some error messages. The **show ntp association detail** printout also indicates whether an NTP peer is successfully authenticated.

You should start the investigation with the **show ntp association detail** printout. This command reports a detailed status of each NTP peer. If you use NTP authentication, the peer status should be *authenticated*. For example, if you have a key mismatch between C1 and C2, the printout will indicate that the peer is not authenticated (see Listing 13).

Listing 13: C1 is not authenticated on C2

```
C2#show ntp associations detail | include 10.2.1.1_

10.2.1.1 configured, insane, invalid, stratum 16
```

A peer could be unauthenticated for a variety of reasons:

- *Key identifier or password mismatch.* The same key identifier and password should be configured on both ends.
- *Missing per-peer key identifier.* In a peering relationship, the key identifier has to be specified on both ends with the **ntp peer key** configuration command.
- *Missing server key identifier.* In a client/server relationship, the key identifier has to be specified on the client with the **ntp server key** configuration command.

If you cannot solve the authentication problems, use the **debug ntp packet peer *ip-address*** command together with the **debug ntp authentication** command to see detailed packet printouts. (A sample printout is included in Listing 11 in the preceding section.)

Once a server or a peer has been authenticated, the router still could refuse to accept time information from it (the remote NTP device would be reported as having *stratum 16*), as illustrated in Listing 14.

Listing 14: Authenticated peer C2 is considered insane on C1

```
C1#show ntp associations detail | include 10.2.1.2_

10.2.1.2 configured, authenticated, insane, invalid, stratum 16

C1#show ntp associations

      address  ref clock     st  when  poll  reach  delay  offset    disp

 ~10.2.1.2    10.2.1.100    16   25    64     4     1.9     3.50  15500.

*~10.2.1.100  127.127.7.1    6   19   512   377    56.2   -5.24     1.2
```

If a router refuses to accept time information from an authenticated peer, the key used by the peer is probably not trusted. This condition is easily verified with the **debug ntp validity** command; packets with untrusted keys generate *Authentication failed* messages, as shown in Listing 15. To fix this error, add the key used by the peer into the list of trusted keys, using the **ntp trusted-key** configuration command.

Listing 15: NTP packet received with an untrusted key

```
04:13:38: NTP: packet from 10.2.1.1 failed validity tests 10

04:13:38: Authentication failed
```

Summary

Once you start relying on your routers having pretty exact time, NTP becomes part of your mission-critical network infrastructure and has to be protected accordingly. Default NTP settings on Cisco IOS allow intruders to change the router's time (or even current year) as soon as the router is not synchronized directly with a primary (stratum 1) NTP server.

To protect the NTP server running on the routers, you should use IP access lists with the **ntp access-group** configuration commands to specify which NTP devices can establish a peering or client/server relationship with the router.

To further protect the validity of the time sources and prevent potential man-in-the-middle attacks, you should configure NTP authentication.

NIL – More Than Just a Training Company

NIL Learning delivers the leading-edge Cisco training to IT professionals and companies around the globe. Through field-proven experts — each both active engineer and instructor — NIL Learning enhances the standard learning curriculum with real-life experience and helps clients to maximize their training investment.

NIL Learning is part of NIL, a leading global IT solutions provider. Since 1992, NIL has been at the forefront of advanced contributors to strategic partner Cisco's technologies, learning curriculum and value-added solutions deployed to clients around the globe. Today, NIL has earned the highest certifications offered by Cisco, VMware, EMC, HP, IBM, Microsoft, F5, Jive, MobileIron, RSA, VCE and others. Their portfolio of solutions consists of managed services, professional services and learning services.

NIL is headquartered in Slovenia, with regional offices in Croatia, Serbia, Saudi Arabia, the U.S., Turkey, South Africa, Morocco, Nigeria, Kenya and Botswana.

Why learn at NIL LEARNING?

- All NIL LEARNING instructors are **field-proven experts** - each both active engineer and instructor.
- **75% of NIL LEARNING engineers hold CCSI** certifica- tions, and **18 have already achieved the respected CCIE rank**.
- NIL LEARNING **enhances the standard learning curriculum** with real-life experience and helps clients to maximize their training investment.
- NIL has been a Cisco Training Partner for many years; it became a **Cisco Learning Partner in 1993**, and has been a **Cisco Gold Partner since 1995**.
- NIL was awarded the **Cisco Most Business Relevant Learning Partner in MEA in 2010 and the most innova- tive learning partner in MEA**.

- NIL received the **Innovation Award for its Technology Led Training and its extensive contribution to Cisco learning solutions** at the Cisco EMEAR Learning Partner Summit in 2012.
- NIL received the **Innovation Award for its Technology Led Training and Advanced Engineer Program** at the Cisco Global Learning Partner Summit in 2013.
- NIL LEARNING runs a **centralized training schedule across the whole EMEAR region**.

More Info

Slovenia

T: +386 1 4746 500

E: sales-support@nil.com

Saudi Arabia

T: +966 1 465 4641

E: info.nilme@nil.com

Botswana

T: +267 318 1684

E: training@it-iq.bw

Serbia

T: +381 11 2282 818

E: info-nilserbia@nil.co.rs

Croatia

T: +385 (0)51 583 255

E: info-nilcroatia@nil.com

South Africa

T: +27 (0)11 575 4637

E: mea_sales@nil.com

Kenya

T: +27 (0)11 575 4637

E: mea_sales@nil.com

Turkey

T: +902 123 81 8639

E: info-nilturkey@nil.com

Morocco

T: +212(0) 660 808 394

E: info-nilmorocco@nil.com

USA

T: +1 612 886 3900

E: info-nilusa@nil.com

Nigeria

T: +27 (0)11 575 4637

E: mea_sales@nil.com

www.learning.nil.com