IPv6 Security Lab - Securing the Router

Basic Best Practice

Getting used to IPv6 show commands in IOS

Try the following IPv6 show commands:

```
show ipv6 interface
show ipv6 neighbors
show ipv6 route
show ipv6 routers
show ipv6 ospf neighbors
show ipv6 ospf rib
show ipv6 traffic
show bgp ipv6 unicast summary
show ipv6 ?
```

The last show command will display all the possible IPv6 status commands on the router. What you see in the list will depend on the IOS in use.

Also, make a note of what each of the above show commands do – the lab instructors may ask later in the exercise.

Setting the time zone on the router

The router can be set with a time zone offset from GMT. The timezone command takes a string of characters – obviously set it to the local timezone. Note that only the first seven characters are used in any time display. The following sets the time zone for IndoChina with a GMT offset of +7.

```
clock timezone ICT 7
```

or

```
clock timezone GMT+7 7
```

Set time stamps for all logs on the router

By default the router will apply only time stamps on log messages from when the router was last powered on. This is not entirely useful in an operational environment, and is a potential security problem, especially when trying to cross compare logs. So we will set time stamps according to the real date and time, and include resolution down to the millisecond:

```
service timestamps debug datetime localtime show-timezone msec
```

```
service timestamps log datetime localtime show-timezone msec
```

Login Banner

IOS by default has a simple welcome message when a new administrative connection to the router is opened. Most Network Operators tend to customise this banner to be appropriate to their business. We will now set up a login banner for the routers in the workshop lab. Use an appropriate greeting – one that doesn't give information away, and makes it very clear that access to the device is restricted to those with permission to do so. If you use an inappropriate greeting, expect the lab instructors to ask you to change it. Use the following example:

```
banner login ^
IPv6 Security Workshop Lab
^
```

Logging

Routers by default capture syslog data produce locally by various features in the IOS. However, the default logging set up is probably not optimal for Network Operators. Each router team should configure logging defaults on their router to be as follows:

```
no logging console
logging source-interface Loopback 0
logging trap debugging
logging buffered 16384 debugging
logging facility local4
```

This command set will set the log source interface to the Loopback 0 interface, trap level to debug (i.e. most detailed), create a 16K buffer on the router and store the most detailed logs there, and any logs sent to the 192.168.1.4 loghost should be sent using syslog facility local4.

It is highly desirable (if not best practice) to disable logging to the router console. If you still haven't done this then the command to do so is no logging console. Console logging is on by default in IOS.

NOTE: For Internet Network Operations, it is strongly recommended to DISABLE console logging – router consoles are usually connected to terminal servers as we have just seen, and if the router is under some stress due to events taking place on it, or on the network, it will waste considerable CPU cycles by sending log messages to the 9600baud console port, thereby further slowing it down. Virtually all Network Operators disable console logging, and alternatively have the syslog messages sent to the local syslog host, as per the configuration above.

Administrative Access

Configuring Telnet VTY access for IPv6

Configure a filter to allow only the trusted hosts to have Telnet access. Note that all attempts are logged by the router system log process, so that there is an audit trail of all access to the router. Part of the AAA suite in Cisco IOS allows these authentication logs to be exported to a syslog server where further access tracking can be undertaken.

```
ipv6 access-list v6-vty-filter
 permit host ipv6-address any
```

Replace *ipv6-address* with the IPv6 address of the **host** you would like to have access. Test this with routers in the same AS; which means that routers in the same AS should permit telnet access from the others. Take the IPv6 address of the physical interface of the adjacent router connecting to your router, and add that into the access-list you have configured.

Now try and include the loopback addresses, as we should normally be telnetting from router to router sourced from the loopback interface. From the address plan, you can see that the loopbacks for the routers in our AS come from 2001:DB8:X0:0::/64 - the first /64 in the IPv6 address block. Update the configuration as per this example:

```
ipv6 access-list v6-vty-filter
 permit 2001:DB8:X0:0::/64 any
```

Applying the filter to the VTY ports

Once the filter is set up, apply it to the vty ports on the router, as in the following example for the Core router on AS101:

```
line vty 0 4
 ipv6 access-class v6-vty-filter in
```

Test to make sure that only telnet from the configured host can have access to the router. To do this, telnet to the adjacent router, and try and telnet back in to yours. You should have ready access. Now telnet to another router in the network and check again – do you get access? If you do, check your filters! Use the debug command to see if you can capture the telnet packets and see the clear-text passwords (be careful with debug!).

Remote Access

Configuring the VTYs for SSH access

Now that we have configured and tested basic IPv6 filtering on the router, we are going to configure access to the router to be somewhat more secure. Everything is sent in the clear through telnet, and it’s use is considered historical and deprecated by most network operators today.

First of all, we need to enable SSH on the routers – SSH is not enabled by default on Cisco IOS. To do this, we first create a domain name for the network:

```
ip domain-name workshop.net
```

and then we need to generate the SSH key pairs for the router:

```
crypto key generate rsa
```

The router will respond, asking what key modulus is required. Choose a modulus of at least 768 (the minimum for SSH version 2), preferably 1024. For example:

```
The name for the keys will be: c1.workshop.net
```

Choose the size of the key modulus in the range of 360 to 4096 for your General Purpose Keys. Choosing a key modulus greater than 512 may take a few minutes. **Let's use 1024 for this lab**:

```
How many bits in the modulus [512]:  1024
% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)
```

Finally set the SSH version to be version 2:

```
ip ssh version 2
```

SSHv2 is now configured on the router, and ready to use. To verify that SSH is working, try and SSH to the router itself:

```
C1# ssh 2001:db8:10::1
Password:
```

```
router1>
```

You should get the password prompt, and following entry of the correct password, the standard command prompt.

Modifying VTY access for SSH

IOS by default allows all transports to connect to the VTY ports. We want to change this behaviour to enhance the security of the router access.

```
line vty 0 4
 ipv6 access-class v6-vty-filter in
 exec-timeout 15 0
transport input ssh
```

This sequence of commands applies the previous configured VTY filter to VTYs 0 through 4, changes the exec-timeout (how long the vty session can remain idle before disconnection) to 15 minutes, and disabling all connecting transports apart from SSH.

Test to make sure that SSH from the permitted host can get access to the router.

Try using telnet from the permitted host as well. Do you get access?

Use the debug command to see if you can capture the SSH packets and see the encrypted passwords.

SSH source address

Most Network Operators use the router Loopback address for administrative purposes as well as the anchor point for their network's iBGP sessions. In this step we will configure SSH so that it uses the loopback interface as the source address for all SSH packets originated by the router.

```
ip ssh source-interface loopback 0
```

To check that this has worked, SSH from your router to a neighbouring router and then enter the "who" command. You will see that you are logged in, and the source address will be displayed.

Disabling Telnet Access

Telnet is a well-known security risk and most network operators disable its use on their network infrastructure equipment. (Equipment which doesn't support SSH should probably be retired and replaced with equipment which supports secure administrator access.) We will now also disable telnet access from the router, as the following example shows:

```
line vty 0 4
 transport output ssh
line con 0
 transport output ssh
```

Try using telnet now for a connection to another router – does it work? What happens?