IPv6 Module 20 – Router Security

Objective: Using an existing dual stack topology, investigate securing the router and the routing infrastructure for the network.

Prerequisites: IPv6 Module 6.

The following will be the common topology used for this supplement.

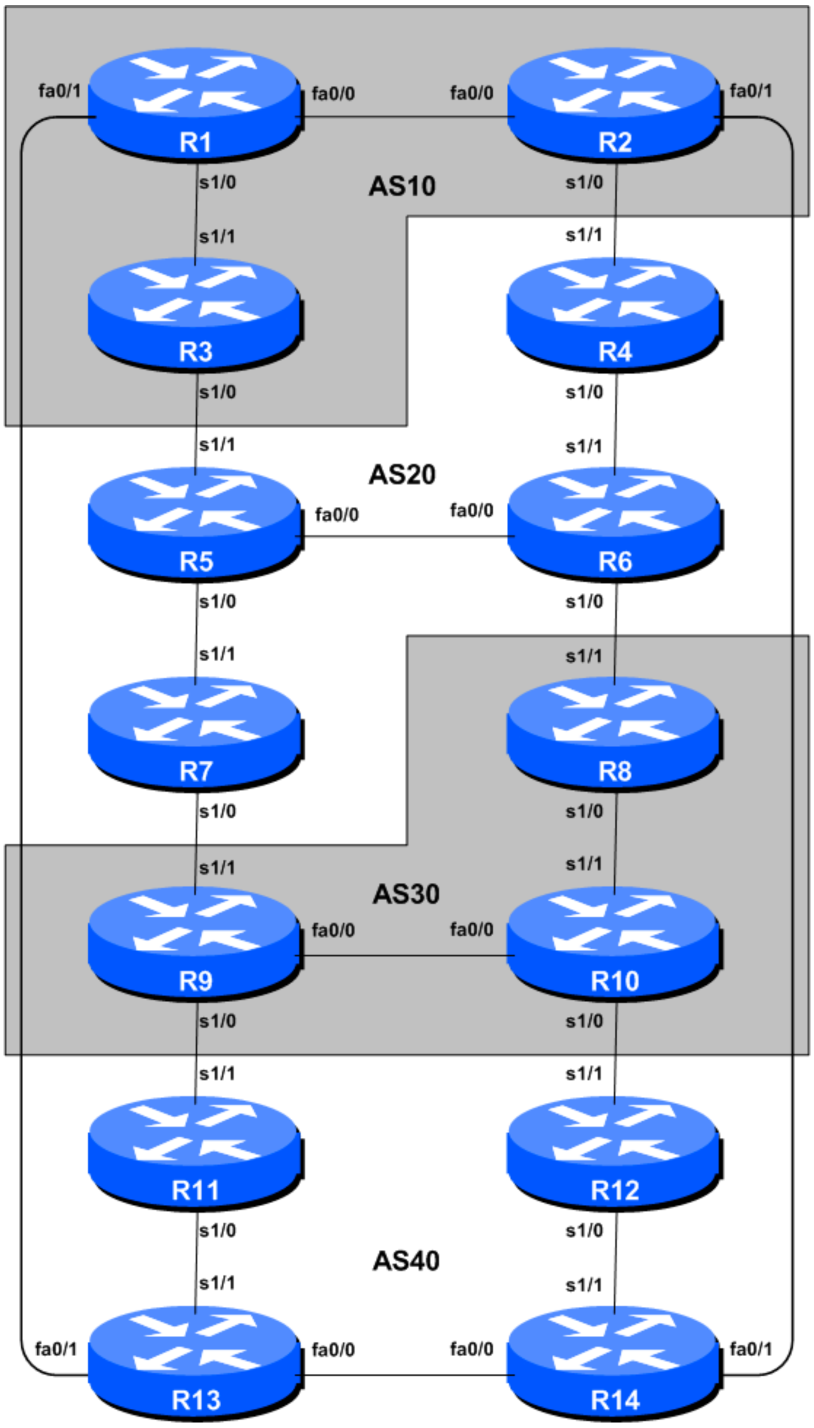

Figure 1 – ISP Lab Basic Configuration

Lab Notes

This Module provides a guide to introducing network security once Module 6 of the IPv6 version of this Workshop series has been complete. The configuration steps described below are aimed at securing the IPv6 portion of the infrastructure – the IPv4 portion can also be secured following a similar procedure.

The routers used for this portion of the workshop must support IPv6. This is basically any IP Plus image from 12.2T onwards (IP Plus was renamed to Advanced IP Services for most platforms as from 12.3 mainline). As always, it is best to check the Cisco Feature Navigator www.cisco.com/go/fn to be absolutely sure which images set and platform supports IPv6. Unfortunately IPv6 is not part of the basic IP only or Service Provider IOS images used by most ISPs.

Note: these labs assume that the routers used are using a minimum of IOS 12.4 mainline. Syntax predating IOS 12.4 is discussed in the optional sections throughout the workshop.

Lab Exercise – Preliminary

1. **Introducing the lab.** If coming from Module 6, this first section can be ignored, as it introduces the participants to accessing the routers.

   This workshop uses Cisco IOS routers running IOS, but on the Dynamips systems – Dynamips translates the PowerPC processor instructions in IOS to those of the host system, allowing Cisco IOS images, and therefore network configurations, to be run on a host PC system (usual Linux or MacOS based).

   The lab will have been preconfigured by the instructors, allowing participants to enter the following exercises directly. Please read the following steps carefully.

2. **Accessing the lab.** The instructors will assign routers to each class group, and will indicate the method of access to the Dynamips server. This will usually be by wireless – if this is the case, make a note of the SSID and any password required. Also make a note of the IP address (IPv4, as Dynamips only supports IPv4 access) of the Dynamips server.

   Access to Dynamips will be by telnet, to a high port, which the instructor will specify. Each participant should ensure that their device has a suitable telnet client. Linux and MacOS system have access to a shell command prompt (or Terminal) programme, which allows telnet at the command line. Windows users can use the Windows "Command Prompt" with the telnet client there, but it's notoriously unreliable. Better to install software such as TeraTerm, HyperTerm or similar third party telnet client.

   Using the client, connect to the router you have been assigned; for example, to connect to the console port of Router 1:

   ```
   telnet 10.0.2.1  2001
   ```

   or to Router 12:

   ```
   telnet 10.0.2.1  2012
   ```

Once connected, you will see the Dynamips response, followed by the login or command prompt of the router:

```
bash-3.2$ telnet 10.0.2.1 2001
Trying 10.0.2.1...
Connected to dynamips.
Escape character is '^]'.
Connected to Dynamips VM "r1" (ID 0, type c7200) - Console port


User Access Verification

Username:
```

If the "Connected to Dynamips VM" won't appear, even after hitting the Return key several times, please request help from the workshop instructors.

3. **Logging into the routers.** Once you have the command prompt as shown in the previous example, you should now log into the routers. The login details are here:

```
Username:   isplab
Password:   lab-PW
Enable:     lab-PW
```

Enter the username and password at the prompt, then type in "enable" to go into configuration mode, and enter the enable password when requested. The command prompt suffix will change from a > to a #. You are now ready to start the first scenario.

Checkpoint #1: *call the lab assistant to verify that you are properly logged in and can use the router. Demonstrate various show commands, for example, "show ip bgp summary" to demonstrate the BGP sessions which are running, and "show ip bgp" to show the BGP routing table.*

Exercise One

4. **Getting used to IPv6 show commands.** Try the following IPv6 show commands, just to get a feel for the IOS CLI. While you demonstrated the BGP commands to the instructor at the first checkpoint, now take the chance to look at the various IPv6 show commands available. Here are a few to try:

```
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.

5. **Set 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 Singapore with a GMT offset of +8.

   ```
   clock timezone SST 8
   ```

   or

   ```
   clock timezone GMT+8 8
   ```

6. **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
   ```

7. **Login Banner.** IOS by default has a simple welcome message when a new administrative connection to the router is opened. Most ISPs tend to customise this banner to be appropriate to their business – see the example given in the ISP Essentials presentation. 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:

   ```
   login banner ^
   ISP Workshop Lab
   ^
   ```

8. **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 ISPs. 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
   logging 192.168.1.4
   ```

   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 ISP 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 ISPs disable console logging, and alternatively have the syslog messages sent to the local syslog host, as per the configuration above.

Exercise Two

9. **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 the these authentication logs to be exported to a syslog server where further access tracking can be undertaken.

```
router1(config)# ipv6 access-list v6-vty-filter
router1(config-ipv6-acl)# permit host ipv6-address any
```

   Replace "ipv6-address" with the IPv6 address of the host you would like to have access. Test this with a physically adjacent router in the class. For example, Router3 could choose either Router1 or Router5. 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.

10. **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 Router1:

```
router1(config)# line vty 0 4
router1(config-line)# ipv6 access-class v6-vty-filter
```

   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!).

Checkpoint #2: *call the lab assistant and demonstrate the function of your telnet VTY filter.*

Exercise Three

11. **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 is 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:

```
router1(config)# ip domain-name workshop.net
```

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

```
router1(config)# 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: router1.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.

How many bits in the modulus [512]:  1024

% Generating 1024 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)

router1(config)#
```

Finally set the SSH version to be version 2:

```
router1(config)# 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:

```
router1# ssh 2001:db8::1
Password:

router1>
```

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

12. **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.

```
router1(config)# line vty 0 4
router1(config-line)# ipv6 access-class v6-vty-filter in
router1(config-line)# exec-timeout 15 0
router1(config-line)# 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.

Checkpoint #3: *call the lab assistant and demonstrate the function of your SSH VTY filter. Also demonstrate that you can no longer use telnet to access your router.*

Exercise Four

13. **Configuring IPv6 Traffic Filters.** We now configure a traffic filter to announce just your subnet (this is BCP38 requirement). For this lab, the address block being used by each AS is in the following table:

AS10	**2001:db8::/32**	**AS30**	**2001:dba::/32**
AS20	**2001:db9::/32**	**AS40**	**2001:dbb::/32**

Each team should set up a traffic filter to only allow traffic from this subnet to exit each router interface. Here is an example for Router1:

```
router1(config)# ipv6 access-list ipv6-netannounce
router1(config-ipv6-acl)# permit ipv6 2001:db8:0::/32 any
router1(config-ipv6-acl)# deny ipv6 any any log
router1(config-ipv6-acl)# exit
router1(config)# interface <interface>
router1(config-if)# ipv6 traffic-filter ipv6-netannounce out
```

Do this for each physical interface on the router which is configured and active. Some routers have two physical interfaces (Serial 1/0 and Serial 1/1), others have three physical interfaces (either two serial and one Ethernet, or one serial and two Ethernet).

14. **Filtering RH Type0 packets.** This configuration required IOS 12.4(2)T or later. Routing Header Type 0 packets have been deprecated by the IETF a few years ago, and are considered a serious security risk. To filter these packets, enter the following configuration:

```
router1(config)# ipv6 access-list deny-sourcerouted
router1(config-ipv6-acl)# deny ipv6 any any routing-type 0
router1(config-ipv6-acl)# permit ipv6 any any
router1(config-ipv6-acl)# exit
```

And then apply the configuration inbound on all physical interfaces which are configured and active on your router. For example:

```
router(config)# ipv6 source-route
router(config)# interface FastEthernet 0/0
router(config-if)# ipv6 traffic-filter deny-sourcerouted in
```

which first turns on support for source routing, and then applies the traffic filter to block RH Type 0.

Most network operators, however, disable all types of Routing Header support, simply by applying the:

```
router(config)# no ipv6 source-route
```

to their basic router security template, as this has already been their practice for many years of IPv4 network operation.

15. **Inbound packet filtering for IPv6 testing.** We will now replace the previous RH0 filter with an access-list which can be used for initial IPv6 testing. It shows had to trap and test for various traffic types running on a router's interface.

```
ipv6 access-list v6starter
  permit icmp any any 2001:db8::/32 echo-reply log-input
  permit icmp any any 2001:db8::/32 echo-request log-input
  permit icmp any any 2001:db8::/32 time-exceeded log-input
  permit icmp any any 2001:db8::/32 packet-too-big log-input
  permit icmp any any 2001:db8::/32 parameter-problem log-input
  permit ipv6 any host <specific host> log-input
  deny ipv6 any any log-input
!
interface <interface>
 ipv6 traffic-filter v6starter in
!
```

Note that the 'log-input' has been included to check what ipv6 traffic is coming in from the outside. Send some ipv6 pings and see if you can see traffic from a 'show log'.

Checkpoint #4: *call the lab assistant and demonstrate the function of your IPv6 starter interface filter. Show the log messages you see from the various testing you have been doing.*

Exercise Five

16. **Configuring IPv4 and IPv6 Routing Security.** Prefix lists allow a network administrator to permit or deny specific prefixes that are sent or received via BGP. Prefix lists should be used where possible to ensure network traffic is sent over the intended paths. Prefix lists should be applied to each eBGP peer in both the inbound and outbound directions.

    Configured prefix lists limit the prefixes that are sent or received to those specifically permitted by the routing policy of a network. If this is not feasible due to the large number of prefixes received, a prefix list should be configured to specifically block known bad prefixes. These known bad prefixes include unallocated IP address space and networks that are reserved for internal or testing purposes by RFC 6890. Outbound prefix lists should be configured to specifically permit only the prefixes that an organization intends to advertise.

    This configuration example uses prefix lists to ensure that no bogon routes are learned or advertised. Create the IPv4 prefix filter named 'bogon-filter' (we will do the same for IPv6 shortly):

```
ip prefix-list bogon-filter deny 0.0.0.0/0
ip prefix-list bogon-filter deny 0.0.0.0/8 le 32
ip prefix-list bogon-filter deny 10.0.0.0/8 le 32
ip prefix-list bogon-filter deny 100.64.0.0/10 le 32
ip prefix-list bogon-filter deny 101.10.0.0/19 le 32
ip prefix-list bogon-filter deny 127.0.0.0/8 le 32
ip prefix-list bogon-filter deny 169.254.0.0/16 le 32
ip prefix-list bogon-filter deny 172.16.0.0/12 le 32
ip prefix-list bogon-filter deny 192.0.2.0/24 le 32
ip prefix-list bogon-filter deny 192.168.0.0/16 le 32
ip prefix-list bogon-filter deny 198.18.0.0/15 le 32
ip prefix-list bogon-filter deny 198.51.100.0/24 le 32
ip prefix-list bogon-filter deny 203.0.113.0/24 le 32
ip prefix-list bogon-filter deny 224.0.0.0/3 le 32
ip prefix-list bogon-filter deny 0.0.0.0/0 ge 25
ip prefix-list bogon-filter permit 0.0.0.0/0 le 32
```

Note the last line – it has a permit statement, allowing the remaining addresses in the BGP session. Cisco IOS has a default deny for its prefix-list filter.

17. **Apply the prefix-filter to eBGP sessions.** We now apply this prefix filter to our external BGP sessions. For example for Router1:

```
router bgp 10
 address-family ipv4 unicast
  neighbor 10.10.15.14 remote-as 40
  neighbor 10.10.15.14 version 4
  neighbor 10.10.15.14 prefix-list bogon-filter in
  neighbor 10.10.15.14 prefix-list bogon-filter out
!
```

Once you have entered the above configuration, refresh the BGP session by entering the following command (example again for Router1). This refresh command applies the newly added BGP configuration to the BGP session.

```
clear ip bgp 10.10.15.14 out
clear ip bgp 10.10.15.14 in
```

Note that it is common for smaller ISPs to have an upstream send only a default route which is enforced via an inbound prefix list. It is also common to create outbound prefix lists to announce only the aggregate allocated prefix outbound to the upstream ISP. The configuration would look something like the following:

```
ip prefix-list DEFAULT-IN permit 0.0.0.0/0
ip prefix-list BGP-OUTBOUND permit <specific aggregate>
!
router bgp 64510
 neighbor 1.2.3.4 remote-as 64509
 neighbor 1.2.3.4 prefix-list DEFAULT-IN in
 neighbor 1.2.3.4 prefix-list BGP-OUTBOUND out
!
```

18. **BGP neighbour authentication (Part 1).** Many network operators implement MD5 protection of their iBGP and eBGP peering sessions. Use of the command is illustrated as follows:

```
router bgp <asn>
 neighbor <ip-address> remote-as <remote-asn>
 neighbor <ip-address> password <secret>
!
```

You will have noticed, if you look at your existing BGP configuration, that MD5 protection of the BGP sessions has already been configured. Partner with one of your eBGP peers and trying removing the password on one of the BGP sessions. For example, for Router1:

```
router bgp 10
 neighbor 10.10.15.14 remote-as 40
 no neighbor 10.10.15.14 password
!
```

What happens to the BGP session?

19. **BGP neighbour authentication (Part 2).** Make sure your neighbour (in this example, Router13), now also removes the password from the BGP session. You'll now see the BGP session re-establish without a password to protect it.

    Now try remove the password encryption protection on the router, by doing:

    ```
    no service password-encryption
    ```

    And now re-enter the passwords for the BGP session. The password to protect the BGP session is simply "cisco". (Do NOT do this in real life on a real network!) Again, for Router1:

    ```
    router bgp 10
     neighbor 10.10.15.14 remote-as 40
     neighbor 10.10.15.14 password cisco
    !
    ```
    Save the configuration of the router ("write memory"), and then display the BGP configuration ("sh conf | b ^router bgp"). Notice how the password for the BGP session is in the clear – you can clearly see the "cisco" in the configuration.

    Now restore password encryption:

    ```
    service password-encryption
    ```

    and you should now see that the password for the eBGP session has been encrypted with Type 7 encryption (not really encryption, it is trivial to reverse engineer, but at least it is no longer readable).

20. **Repeat with IPv6.** Each Router Team should now repeat the previous steps above using IPv6 instead. First off we will create the special use IPv6 prefix-list:

    ```
    ipv6 prefix-list v6bogon-filter permit 2001::/32
    ipv6 prefix-list v6bogon-filter deny 2001::/32 le 128
    ipv6 prefix-list v6bogon-filter deny 2001:db8::/32 le 128
    ipv6 prefix-list v6bogon-filter permit 2002::/16
    ipv6 prefix-list v6bogon-filter deny 2002::/16 le 128
    ipv6 prefix-list v6bogon-filter deny 3ffe::/16 le 128
    ipv6 prefix-list v6bogon-filter permit 2000::/3 le 48
    ipv6 prefix-list v6bogon-filter deny ::/0 le 128
    ```

    and then apply it to our IPv6 eBGP session (or sessions) with our neighbours, in the same style as we did for IPv4. Here is the Router1 to Router13 eBGP session example:

    ```
    router bgp 10
     address-family ipv6 unicast
      neighbor 2001:DB8:0:4::1 remote-as 40
      neighbor 2001:DB8:0:4::1 version 4
      neighbor 2001:DB8:0:4::1 prefix-list v6bogon-filter in
    ```

```
  neighbor 2001:DB8:0:4::1 prefix-list v6bogon-filter out
!
```

Don’t forget to refresh the BGP session after applying the filter. For example:

```
clear bgp ipv6 uni 2001:db8:0:4::1 in
clear bgp ipv6 uni 2001:db8:0:4::1 out
```

Checkpoint #5: *call the lab assistant and demonstrate the IPv4 and IPv6 BGP filters as applied to your eBGP sessions.*

Exercise Six

21. **Exploring Netflow for IPv4 and IPv6.** Netflow identifiesFlow identifies anomalous and security-related network activity by tracking network flows. NetFlow data can be viewed and analysed via the command line interface (CLI), or the data can be exported to a commercial or freeware NetFlow collector for aggregation and analysis. NetFlow collectors, through long-term trending, can provide network behavior and usage analysis. NetFlow functions by performing analysis on specific attributes within IP packets and creating flows. Version 5 is the most commonly used version of NetFlow, however, version 9 is more extensible and is required to support IPv6. NetFlow flows can be created using sampled traffic data in high-volume environments. Cisco Express Forwarding (CEF) is a prerequisite to enabling NetFlow.

    NetFlow can be configured on routers and switches. In older releases of Cisco IOS software, the command to enable NetFlow on an interface was:

    ```
    ip route-cache flow
    ```

    In newer releases of Cisco IOS (12.4 onwards), the command has been replaced by:

    ```
    ip flow {ingress | egress}
    ```

    The following configuration illustrates the basic configuration of this feature.

    ```
    ip flow-export destination <ip-address> <udp-port>
    ip flow-export version <version>
    !
    interface fastethernet 0/0
     ip flow ingress
     ip flow egress
    !
    ```

    The following is an example of NetFlow output from the router command line interface. The SrcIf attribute can aid in traceback.

```
router#show ip cache flow
IP packet size distribution (26662860 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .741 .124 .047 .006 .005 .005 .002 .008 .000 .000 .003 .000 .001 .000 .000

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .000 .000 .001 .007   .039   .000   .000  .000  .000  .000   .000

IP Flow Switching Cache, 4456704 bytes
```

```
  55 active, 65481 inactive, 1014683 added
  41000680 ager polls, 0 flow alloc failures
  Active flows timeout in 2 minutes
  Inactive flows timeout in 60 seconds
IP Sub Flow Cache, 336520 bytes
  110 active, 16274 inactive, 2029366 added, 1014683 added to flow
  0 alloc failures, 0 force free
  1 chunk, 15 chunks added
  last clearing of statistics never

Protocol       Total    Flows   Packets  Bytes  Packets Active(Sec) Idle(Sec)
--------       Flows    /Sec     /Flow   /Pkt     /Sec     /Flow     /Flow
TCP-Telnet     11512     0.0        15     42      0.2      33.8      44.8
TCP-FTP         5606     0.0         3     45      0.0      59.5      47.1
TCP-FTPD        1075     0.0        13     52      0.0       1.2      61.1
TCP-WWW        77155     0.0        11    530      1.0      13.9      31.5
TCP-SMTP        8913     0.0         2     43      0.0      74.2      44.4
TCP-X            351     0.0         2     40      0.0       0.0      60.8
TCP-BGP          114     0.0         1     40      0.0       0.0      62.4
TCP-NNTP         120     0.0         1     42      0.0       0.7      61.4
TCP-other     556070     0.6         8    318      6.0       8.2      38.3
UDP-DNS       130909     0.1         2     55      0.3      24.0      53.1
UDP-NTP       116213     0.1         1     75      0.1       5.0      58.6
UDP-TFTP         169     0.0         3     51      0.0      15.3      64.2
UDP-Frag           1     0.0         1   1405      0.0       0.0      86.8
UDP-other      86247     0.1       226     29     24.0      31.4      54.3
ICMP           19989     0.0        37     33      0.9      26.0      53.9
IP-other         193     0.0         1     22      0.0       3.0      78.2
Total:       1014637     1.2        26     99     32.8      13.8      43.9

SrcIf   SrcIPaddress      DstIf     DstIPaddress     Pr    SrcP     DstP   Pkts
Gi0/1   192.168.128.21     Local    192.168.128.20   11    CB2B     07A       3
Gi0/1   192.168.150.60     Gi0/0    10.89.17.146     06    0016     101F     55
Gi0/0   10.89.17.146       Gi0/1    192.168.150.60   06    101F     0016      9
Gi0/1   192.168.150.60     Local    192.168.206.20   01    0000     0303     11
Gi0/0   10.89.17.146       Gi0/1    192.168.150.60   06    07F1     0016      1
```

Repeat netflow configuration to work with IPv6. Hint: the commands are very similar!

Remember to use '?' if you want to check where you can configure it. Also, remember that most IPv6 configurations in cisco devices start with 'ipv6' to denote this is for IPv6 traffic.

<u>Checkpoint #6:</u> *call the lab assistant and demonstrate what you have been able to catch by using Netflow on your router. Show both the IPv4 and IPv6 netflow output.*