PostLab 10: Digging into IPv4 & IPv6 ACLs

What You Will Do:

1. Start practicing for the SBA by setting up IPv4 and IPv6 infrastructure: DHCP, OSPF
2. Ensure you are familiar with using a L3 switch as a router (some review, some new?)
3. Ensure you are familiar with stateful DHCPv6 config required for: server, interface, client
4. Enable multiple services on a Cisco device: telnet, SSH, HTTP, HTTPS
5. Review Lab 10 by implementing ACLs for IPv4 access to those services
6. Implement IPv6 ACLs for access to the same services over IPv6

Things that you will need to know or learn:

1. Convert switched interfaces on a Switch to routed interfaces (review from NET1006)
2. How to set up OSPFv2 and OSPFv3 (repeat of previous labs)
3. The basic structure of Cisco ACLs has 4 parts and is very consistent for both IPv4 & IPv6:
   [Yes/No] [proto: ip/tcp/udp] [Src identification [and port]] [Dst identification [and port]] [log]

What you need to submit and when:

1. Complete post-lab exercise and quiz on BrightSpace, **before** your next lab.

This post-lab is worth **36%** of this lab, even though the number of points may differ between the three parts.

Required Equipment:

- A laptop or desktop computer, with telnet capability
- Access to Algonquin's NetLab facility

References and Resources:

- **Appendices** for specific tasks (review from previous courses or repeat from this course)
- **Solutions** to Lab 10 - Prelab (posted on course site)
- Slide deck on ACLs for IPv4 and IPv6, posted on the course site
- lecture summary notes - Wk09 and Wk10

If you run into problems, please

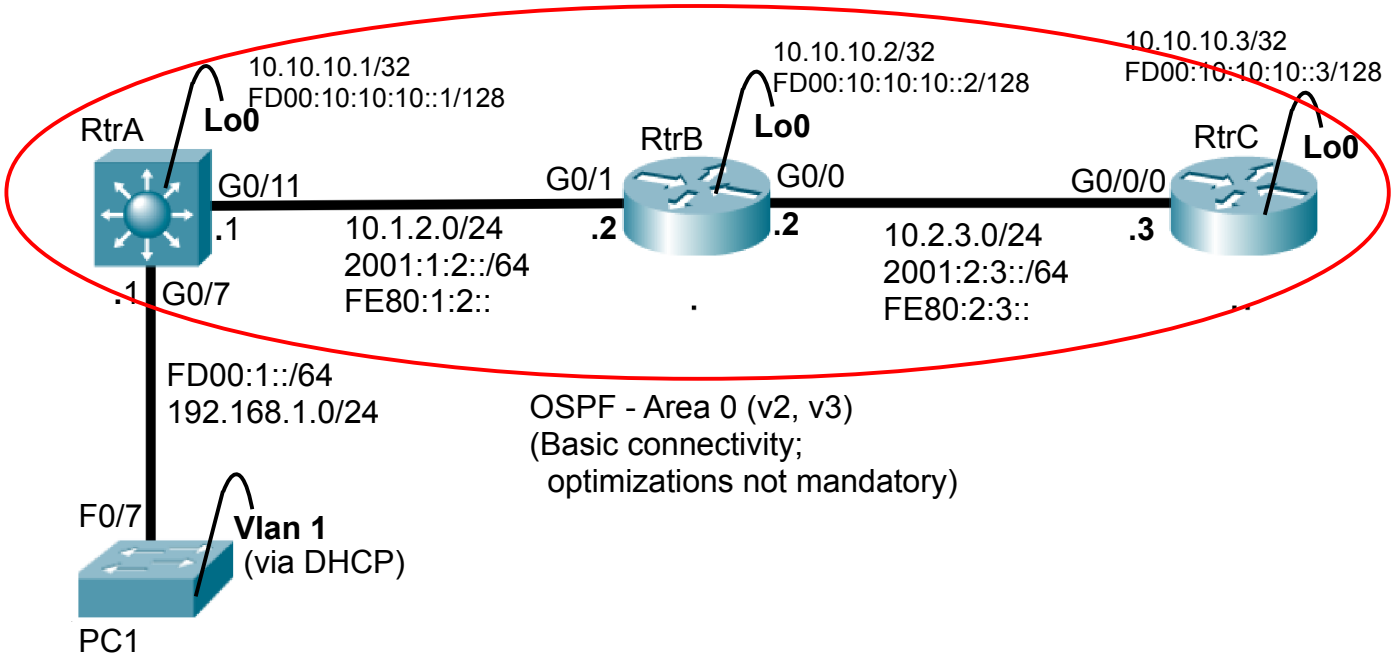
READ THE INSTRUCTIONS

and

CHECK THE APPENDICES

Everything in this exercise is here for a reason!

Topology and Addressing Diagram



Device	Device	Device
RtrA (3560-X) (left side = S1, right side = S2)	**RtrB** (2911) (left side = R1, right side = R3	**RtrC** (4331) (lft side = R2, rght side=R4)
RPC1 (2960) (left side = S3, right side = S4)		

Task 1: Review the Lab exercise instructions

You'll need a booking for a right or left 1/2 pod on NetLab. Confirm your background knowledge:
– To set enable IPv6 on a switch, see Appendix A.
– On a switch convert an interface to routed mode, and enable IPv4 routing, see Appendix B.
– To configure DHCPv4 and DHCPv6 **clients** and servers, see Appendix C.

Task 2: Configure and Verify the underlying infrastructure

Configuring the infrastructure should all be review. Each item should take 1-10 mins maximum.

– Enable IPv6 routing for all routers (and the L3 switch!). Enable IPv4 routing on the L3 switch.
– Convert RtrA G0/7, G0/11 to routed mode.
– Configure all interfaces with their IPv4 and IPv6 addresses (link-local + routable) and 'no shut'!
– Configure DHCP servers for the "PC" subnet.
– Configure PC1 (Vlan 1 interface) address as *dhcp* for both IPv4 and IPv6.
– Configure OSPFv2 and OSPFv3; include **all** interfaces & loopbacks for all **3** routers.

Verify that you have full IPv4 and IPv6 connectivity between the PC and all router loopbacks. Everything **must** be working or you'll have no hope of getting correct results for ACLs!

Q. Confirm/prove that you converted switch ports to routed mode: in 'sh ip int xxx', what is listed for the broadcast address?

Q. Confirm/prove that DHCPv4 worked: on RtrA show the the dhcp bindings (Appendix C). For the PC IP address, what is the exact value in the column "Type"?

Q. Confirm/prove that DHCPv6 worked: on the "PC" (2960 switch), show the IPv6 routing table. For the default route, what is the protocol code (i.e. first column)? For the subnet on that segment, what is the protocol code? (=

Q. Did you include all interfaces in OSPF? Confirm/prove it by correctly answering: How many OSPFv2 routes on RtrC? How many OSPFv3 routes on RtrC?

Task 3: Configure services

In preparation for testing the ACLs, you'll configure four services on each router: telnet, SSH, HTTP, HTTPS. Ping is enabled by default, so there's no need configure it!

Now think about this for a moment: Do you need to configure these services separately for IPv4 and IPv6? What OSI layer are they: L3 (transport) or another layer? If they're not L3, then the method used to reach them is irrelevant and no separate config is needed for each IP protocol.

– Hopefully, you remembered all the basic config for a network device (including hostame!)
– If you haven't done it already, you should be able to copy & paste from previous labs.
– Your basic config from previous labs should already include telnet & SSH; if not, add it.
– Add any additional config required for the services to work with IPv6.
– From "PC1" **verify** the telnet and SSH services on each of the 3 routers.
Proceed with the next task only when all services are working correctly.

Q. What specific additional config is necessary to make each of these services work with IPv6? (i.e. Other than the *ipv6 unicast-routing* command to enable IPv6 overall.)

(Continued on next page)

Task 4: Using Telnet as a Tool for Verifying Services

Yes, telnet is to another incredibly useful tool (like ping & traceroute) that can be used to verify any service such as HTTP. We'll have a lot more interesting interaction if the connection is unencrypted and we know the syntax of the application protcol (or "language").

We can use this tool as a quick, effective test for reachability when verifying ACLs.

Below are samples of 2 successful and 2 unsuccessful connections. Verify that you can reach the HTTP and HTTPS servers on the PC and RtrA from Rtr C.

Success #1 - HTTP

```
PC1# telnet  10.10.10.1  80
Trying 10.10.10.1, 80 ... Open
GET / HTTP/1.0

HTTP/1.1 401 Unauthorized
Date: Mon, 02 Jan 2006 00:11:40 GMT
Server: cisco-IOS
Accept-Ranges: none
WWW-Authenticate: Basic realm="level_15_access"

401 Unauthorized
[Connection to 10.10.10.1 closed by foreign host]
```

! Connection initiated to port 80
SUCCESS with 3-way handshake!
! There's **NO PROMPT**, just **START TYPING**; and hit enter **TWICE**

In HTTP "language", this requests the root page ("/") using the version of HTTP requiring no hostname

The server *did* respond: it said "you are not authorized", but that is a response!

Success #2 - HTTPS

```
RtrA# telnet 192.168.1.20  443
Trying 192.168.1.20, 443 ... Open
hello
[Connection to 192.168.1.20 closed by foreign host]
RtrA#
```

SUCCESS with 3-way handshake!
No encryption so server quits, but we *did* get a connection

Unsuccessful #1 - No server / invalid port

```
RtrA# telnet 192.168.1.20 888
Trying 192.168.1.20, 888 ... _____
% Connection refused by remote host
RtrA#
```

No server listening on the port, so we see the device actively rejecting the SYN. It at least proves we got through to the device.

Unsuccessful #2 - TCP SYN can't reach the server

```
RtrA#telnet 127.0.0.1 80
Trying 127.0.0.1, 80 ... _____
% Connection timed out; remote host not responding
```

Didn't even reach the device. ACLs blocking access is one way this could happen!

Q. What 3-digit HTTP code do you get if you request the page stats.html?
(the request should be: GET /stats.html HTTP/1.0 followed by <Enter> *twice*)

Task 5: Deploy an IPv4 ACL

Review and practice the skills from the main lab. You'll need to:
– We want to control or limit traffic from all "PCs" to Rtr**C**
– Compose an ACL to permit ICMP, telnet, SSH, HTTP, HTTPS; explicitly block all other traffic
– The ACL must be placed in the **IN**bound direction on Rtr**B** (int G0/1) or Rtr**C** (int G0/0/0).
– Be as specific as possible for the PCs, given DHCP means we don't know exact host addrs
– Since the interface is routed, the match counters *will* increment, so no need for logging!
– It will be *much easier* to edit the ACL if you include line numbering.

Step 1. Double check your ACL to make sure that you didn't forget anything!
Step 2. Enter your ACL, then apply it to Rtr**B** in the **in**bound direction on interface G0/1.
Step 3. If you followed the instructions **accurately**, you can probably ping RtrC (quickly!)
Step 4. Verify that you can telnet from the PC to each of the routers.
Keep an eye on the Rtr**B** console while you do so.
Step 5. Do you see any syslog messages on RtrA or RtrB console? If yes, copy and save the exact message. If not, continue testing telnet (e.g. from the PC to RtrB).
Step 6. Verify full connectivity between loopbacks: RtrA<—>RtrB, RtrB <—> RtrC.
Test with both IPv4 and IPv6. What's the status of OSPFv2 & OSPFv3 neighbours?
Record and save the output of the show commands.
Step 7. Did the ACL cut off one protocol too many? Perhaps a routing protocol that was active? Check the slide deck on IPv4 ACLs to figure out how to allow that protcol.
Step 8. Add a line to your ACL (you did use numbers, didn't you?) to allow that protocol.
Step 9. Wait a handful of seconds (or ~30 secs if your interfaces are still multi-access).

Q. What is the exact message that appears on RtrB's console?
Q. On Rtr**A** what is the *state* of the OSPFv2 neighbour, and OSPFv3 neighbour?
Q. What is the absolute simplest form of extended ACE that restores full connectivity?

Task 6: Create and Deploy an IPv6 ACL

Converting an ACL from IPv4 to IPv6 involves a very limited number of steps. We'll start by exploring some behaviors of Cisco's IOS, then convert the ACL with only 3 main changes required. Note that ACE sequence numbers are handled *much better* for IPv6, so it's completely unnecessary to enter them manually.

Step 1. Are ACL names unique or common to IPv4 and IPv6? Attempt to create an IPv6 ACL with the exact same name as your IPv4 ACL. What's the message you get?
Step 2. Continue with starting the creation of an IPv6 ACL, modifying as necessary.
Step 3. Start an ACE and get some help about protocols: **`permit ?`**
Do you see OSPF listed, as you would see in the IPv4 ACL help?
Step 4. Ok, use a text editor to convert your IPv4 ACL into IPv6:
– For ICMP, just do a direct copy of the simplest possible ACL: **`permit icmp any any`**
– It may help to know the protocol number for OSPF is 89: **`permit 89 any any`**
– Replace any IPv4 addresses (192.168.1.0/24) with the IPv6 address: **`FD00:1::/64`**
– Modify the final deny statement to fit IPv6: **`deny ipv6 any any`**
... DONE!
Step 5. Apply the IPv6 ACL **in**bound on interface G0/1 in RtrB. Verify with multiple ping, telnet, SSH, and HTTP tests that you still have full connectivity!

Q. What message do you get if an IPv4 and IPv6 ACL have the same name?
Q Does the IPv6 ACL help for protocols include a specific item for OSPF?
Q. Search the internet: what is the protocol number for OSPFv2 and OSPFv3?
Are they the same or different ?

Task 7: Backups and Clearing the equipment

The next postlab uses the same topology so you'll want backups of all your devices.
Please leave the SDM configuration at dual-ipv4-ipv6, but please ensure your equipment has no saved config (i.e. no startup-config).

Though you may not have memorized everything new in this lab, hopefully you have become aware of several new aspects of IPv6, ACLs, and peculiarities of Cisco network devices.

Welcome to the career you have chosen with the BIT-NET degree. :-)

Appendix A - Enabling IPv6 on Cisco switches

Until fairly recently (~5 years ago?), Cisco switches were ***not*** capable of using IPv6 addressing in their default configuration. This applies to the 2960 series switches that we have in both T108 and NetLab. Fortunately it's easy to change the setting which allows IPv6 addresses, followed by a reload; you can think of this a similar to changing a BIOS setting on a PC.

To ensure you don't spend time needlessly, it makes sense to first check whether the change is necessary and whether it has already been made. Use the command **`show sdm prefer`** and then look for the ability to have IPv6 multicast routes. If that number is 0, then it's necessary to change the SDM setting: **`sdm prefer dual-ipv4-and-ipv6 default`** in global config mode and then do a reload (please, no need to save config changes).

Below are command sequences from a 3560-X switch (no need to change the setting from default) and a 2960 (change needed and implemented).

```
S1#sh sdm prefer
 The current template is "desktop default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 8 routed interfaces and 1024 VLANs.

  number of unicast mac addresses:                  6K
  number of IPv4 IGMP groups + multicast routes:    1K
  number of IPv4 unicast routes:                    8K
    number of directly-connected IPv4 hosts:        6K
    number of indirect IPv4 routes:                 2K
  number of IPv6 multicast groups:                  64
  number of IPv6 unicast routes:                    106
    number of directly-connected IPv6 addresses:    74
    number of indirect IPv6 unicast routes:         32
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 0.875k
  number of IPv6 policy based routing aces:         0
  number of IPv6 qos aces:                          0
  number of IPv6 security aces:                     60
```

Ok, no SDM change needed on this 3560-X

```
S3(config)#do sh sdm prefer
 The current template is "default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 0 routed interfaces and 255 VLANs.

  number of unicast mac addresses:                  8K
  number of IPv4 IGMP groups + multicast routes:    0.25K
  number of IPv4 unicast routes:                    0
  number of IPv6 multicast groups:                  0
  number of IPv6 unicast routes:                    0
    number of directly-connected IPv6 addresses:    0
    number of indirect IPv6 unicast routes:         0
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.125k
```

On this 2960, the SDM needs to be changed

```
  number of IPv4/MAC security aces:                 0.375k
  number of IPv6 policy based routing aces:         0
  number of IPv6 qos aces:                          20
  number of IPv6 security aces:                     25

S3(config)#sdm prefer ?
  default             Default bias
  dual-ipv4-and-ipv6  Support both IPv4 and IPv6
  lanbase-routing     Supports both IPv4 and IPv6 Static Routing
  qos                 QoS bias

S3(config)#sdm prefer dual
S3(config)#sdm prefer dual-ipv4-and-ipv6 ?
  default  Default bias
  vlan     VLAN bias

S3(config)#sdm prefer dual-ipv4-and-ipv6 default
Changes to the running SDM preferences have been stored, but cannot take
effect
until the next reload.
Use 'show sdm prefer' to see what SDM preference is currently active.
S3(config)#do reload

System configuration has been modified. Save? [yes/no]: no
Proceed with reload? [confirm]
[...switch reboots...]
S3(config)#do sh sdm prefer
 The current template is "dual-ipv4-and-ipv6 default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 0 routed interfaces and 255 VLANs.

  number of unicast mac addresses:                  4K
  number of IPv4 IGMP groups + multicast routes:    0.25K
  number of IPv4 unicast routes:                    0
  number of IPv6 multicast groups:                  0.375k
  number of IPv6 unicast routes:                    0
    number of directly-connected IPv6 addresses:    0
    number of indirect IPv6 unicast routes:         0
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.125k
  number of IPv4/MAC security aces:                 0.375k
  number of IPv6 policy based routing aces:         0
  number of IPv6 qos aces:                          0.625k
  number of IPv6 security aces:                     125

S3(config)#int vlan 1
S3(config-if)#ip?
ip  ipv6

S3(config-if)#
```

Yes, IPv6 IS enabled on the switch

Appendix B - Converting switched interfaces on a Switch to routed interfaces

Normally on a switch, all interfaces (within the same VLAN) act as a common shared segment. Most significantly, interfaces don't and can't have individual IP addresses. We also learned this week that switched interfaces on a Cisco box can only have ACLs applied **in**bound, and they don't increment the hardware match counters.

We don't see these limitations on router interfaces, and fortunately, the limitations disappear if interfaces on a switch are converted to routed mode. When a switch has the ability to selectively flip interfaces between routed and switched modes, the switch is known as a *L3 switch*, as opposed to a pure *L2 switch*. Cisco series 3560 and higher models are L3 switches.

In case you're wondering, Cisco *routers* do *not* have the ability to flip interfaces into switched mode.

Setting a switch interface to routed mode takes exactly one command: `no switchport`
After that you can treat it exactly as you would any router interface.

Appendix C - DHCP server configuration for IPv4 and IPv6

Students have indicated the DHCP server configuration has been covered in a previous course. Stateful DHCPv6 was used in an IPv6 lab earlier in the semester. For convenience, sample config for both versions of DHCP are given below. It can be used on routers and L3 switches.

```
CiscoL3Device(config)# service dhcp     ! For good measure / anti-bugging
CiscoL3Device(config)# ip dhcp pool  PC_LAN
CiscoL3Device(dhcp-config)# network  4.3.2.0 255.255.255.0
CiscoL3Device(dhcp-config)# default-router  4.3.2.1
CiscoL3Device(dhcp-config)# domain-name bitnet.com
CiscoL3Device(dhcp-config)# dns-server  1.1.1.1
```

IPv4 SERVER config

```
CiscoDHCP-Client(config-if)# ip address dhcp
```

IPv4 CLIENT config

For IPv4, the pool is automatically associated with the interface that has an address is within the DHCP pool subnet.

No IPv4 INTERFACE config

Verify addresses that have been leased with: **`show ip dhcp binding`**

Look for the parallels and differences with (stateful) DHCPv6, including the requirement for interface configuration specifying the DHCP mode and address pool to be used.

```
CiscoL3Device(config)# ipv6 unicast-routing     ! Don't forget this
CiscoL3Device(config)# ipv6 dhcp pool STATEFUL-DHCP
CiscoL3Device(config-dhcp)# address prefix 2001:4:3:2::/64
CiscoL3Device(config-dhcp)# dns-server 2606:4700:4700::1111    ! CloudFlare's IPv6 DNS
CiscoL3Device(config-dhcp)# dns-server 2001:4860:4860::8888    ! Google's IPv6 DNS
CiscoL3Device(config-dhcp)# domain-name bitnetipv6.com
```

IPv6 SERVER config

```
CiscoL3Device(config)# interface g0/1
CiscoL3Device(config-if)# ipv6 nd managed-config-flag
CiscoL3Device(config-if)# ipv6 dhcp server STATEFUL-DHCP
```

IPv6 INTERFACE config

```
CiscoDHCP-Client(config-if)# ipv6 address autoconfig
```

IPv6 CLIENT config

Verify addresses that have been leased with: **`show ipv6 dhcp binding`**