Faculty of Engineering and Technology
Electrical and Computer Engineering Department
ENCS4130 – Computer Network Laboratory

Experiment No. 09

Access Control Lists (ACLs)

■ Objectives

(a) Understand how to configure and verify access control lists on Cisco routers.

(b) Learn the differences between standard and extended ACLs.

■ Requirements

(a) Cisco Packet Tracer software.

(b) Two routers.

(c) Three standard switches and one multilayer switch.

(d) Seven PCs, two laptops, and one server.

■ Pre-Lab Task

Students are required to build the network topology shown in Figure 1 and establish full connectivity by completing the steps outlined in Section 9.2.1.

9.1 Introduction

An **access control list (ACL)** is a set of rules used to control network traffic and enhance security. Each rule in an ACL is referred to as an *access control entry*. Although the full term is commonly used in formal documentation, network professionals often refer to them simply as *access lists*.

ACLs operate by *filtering network traffic* at router interfaces, determining whether packets should be forwarded or dropped. When a packet reaches an interface, the router evaluates it against the ACL to decide whether to ***permit*** or ***deny*** it. This evaluation is based on defined criteria such as *source IP address*, *destination IP address*, *protocol type*, and *port number/name*.

9.1.1 Objectives of ACL Implementation

ACLs are implemented for a variety of purposes, including:

- **Enhancing Network Security (Primary Focus of This Experiment):** ACLs act as a fundamental security mechanism by controlling access to network resources. By selectively allowing or blocking traffic, they function as a basic firewall, helping prevent unauthorized access, enforce security policies, and minimize exposure to potential threats.
- **Controlling Network Traffic Flow:** ACLs enable administrators to manage traffic based on defined parameters, allowing them to optimize bandwidth usage and improve overall network performance.
- **Filtering Routing Updates:** ACLs can be used to control which routing updates are accepted or advertised by a router. This is beneficial in limiting unnecessary routing information propagation, thereby increasing the efficiency and stability of network routing.

9.1.2 ACL Identifiers on Cisco Devices

Cisco devices support two types of ACL identifiers:

- **Named ACLs:** These use descriptive text-based names (e.g., `BZU`) for better readability and easier management.
- **Numbered ACLs:** These use numeric identifiers (e.g., `10`, `105`) and are commonly used in simple or small-scale implementations.

Note:

This experiment will utilize **numbered ACLs**.

9.1.3 Wildcard Masks

Wildcard masks—also referred to as ***inverse masks***—are used in ACLs to specify which bits of an IP address must match during evaluation:

- A bit value of `0` indicates that the corresponding bit ***must match exactly***.
- A bit value of `1` indicates that the corresponding bit ***can be any value***.

This logic is the inverse of subnet masking. For example:

- Subnet mask: `255.255.255.224`
- Corresponding wildcard mask: `0.0.0.31`

Wildcard masks are essential for accurately defining the range or scope of IP addresses to which an ACL rule applies.

9.1.4 ACL Configuration

To configure an ACL, the administrator must define one or more permit or deny conditions. All statements within a given ACL must share the same unique ACL number. Each entry defines a specific type of traffic to be filtered.

ACL entries are ***processed sequentially***, from top to bottom. The first matching rule determines how the packet is handled. Therefore, the ***order of ACL statements is critical*** to achieving the desired traffic control behavior.

A) Standard ACLs

Standard ACLs filter traffic ***only by source IP address***. They do not consider destination IP addresses, protocol types, or port numbers. Standard ACLs are typically identified using numbers in the **range 1–99**.

General Syntax:

```
Router(config)# access-list <access-list-number>
                <permit|deny>
                <host src-addr|src-addr src-wildcard|any>
```

For example, to block all traffic from the `192.63.1.0/24` network:

```
Router(config)# access-list 10 deny 192.63.1.0 0.0.0.255
```

Once the ACL is defined, it must be applied to a specific interface in either the *inbound* or *outbound* direction.

Applying an ACL to an Interface:

```
Router(config)# interface <interface-name>
Router(config-if)# ip access-group <access-list-number>
                   <in|out>
```

For example, to apply **ACL 10** to `FastEthernet 0/0` in the *inbound* direction:

```
Router(config)# interface fa0/0
Router(config-if)# ip access-group 10 in
```

B) Extended ACLs

Extended ACLs provide ***more granular control*** by allowing traffic filtering based on:

- Source and destination IP addresses.

- Transport-layer protocols (e.g., TCP, UDP, ICMP).
- Specific port numbers or port names.

Extended ACLs use numbers in the **range 100–199**.

General Syntax:

```
Router(config)# access-list <access-list-number>
                <permit|deny> <protocol>
                <host src-addr|src-addr src-wildcard|any>
                <host dest-addr|dest-addr dest-wildcard|any>
                eq <port>
```

For example, to deny HTTP traffic (TCP port 80) from host `192.63.1.2` to web server `172.16.100.100`:

```
Router(config)# access-list 120 deny tcp host 192.63.1.2
                host 172.16.100.100 eq 80
```

Or, using a ***well-known port name*** instead of the number (Table 1 provides examples of supported names):

```
Router(config)# access-list 120 deny tcp host 192.63.1.2
                host 172.16.100.100 eq www
```

Note:

If the **eq 80** (or **eq www**) is **omitted**, the ACL will block **all TCP traffic** between the two hosts, potentially affecting other services such as FTP or Telnet.

Table 1: Common port names used in extended ACLs.

Port Name	Port Number	Protocol	Service
`www`	80	TCP	HTTP
`https`	443	TCP	HTTPS
`ftp`	21	TCP	FTP Control
`ftp-data`	20	TCP	FTP Data
`telnet`	23	TCP	Telnet
`smtp`	25	TCP	Simple Mail Transfer Protocol
`pop3`	110	TCP	Post Office Protocol v3
`imap`	143	TCP	Internet Message Access Protocol
`domain`	53	TCP/UDP	Domain Name System (DNS)
`ssh`	22	TCP	Secure Shell (SSH)

After defining an extended ACL, apply it to the relevant interface and direction. For example, to apply **ACL 120** to `FastEthernet 0/1` in the *outbound* direction:

```
Router(config)# interface fa0/1
Router(config-if)# ip access-group 120 out
```

9.1.5 Implicit "Deny All" Rule

At the end of every ACL, ***Cisco implicitly appends a "deny all traffic" rule***. This means that *any traffic not explicitly permitted will be denied by default*. To avoid unintended disruptions, it is recommended to include a *general permit statement* at the end of the ACL if necessary.

Consequently, a complete ACL that blocks only HTTP traffic (TCP port 80) from host `192.63.1.2` to web server `172.16.100.100` is as follows:

```
Router(config)# access-list 120 deny tcp host 192.63.1.2 host
                172.16.100.100 eq 80
Router(config)# access-list 120 permit ip any any
```

> Important:
>
> Removing a single rule using the **no** keyword (e.g., **no access-list 120 ...**) will **remove the entire ACL**, not just the specified entry. Always re-define all required rules after making changes.

9.1.6 Viewing ACLs

To display the current ACL configuration and traffic match statistics, use the following command:

```
Router# show access-lists
```

9.2 Procedure

In this lab, we will connect two routers and multiple PCs distributed across different networks and virtual local area networks (**VLANs**). VLAN 40 will be configured on **Router1**, while VLANs 50 and 60 will be set up on **Multilayer Switch0**. To enable communication between subnets and VLANs, we will implement the open shortest path first (**OSPF**) routing protocol on both the routers and the multilayer switch. After establishing connectivity, we will configure **ACLs** to control and restrict traffic flows between specific users and networks, demonstrating practical use of traffic filtering in a segmented and dynamically routed network.

9.2.1 Building the Topology and Preparing Full Connectivity

Construct the network topology shown in Figure 1, which consists of the following devices:

- Routers: Use **Router-PT**.
- Multilayer Switch: Use **3560-24PS**.
- Switches: Use **Switch-PT**.
- Servers: Use **Server-PT**.
- PCs/Laptops: Use **PC-PT/Laptop-PT**.
- Connections: Use the "**Automatically Choose Connection Type**" to connect devices.

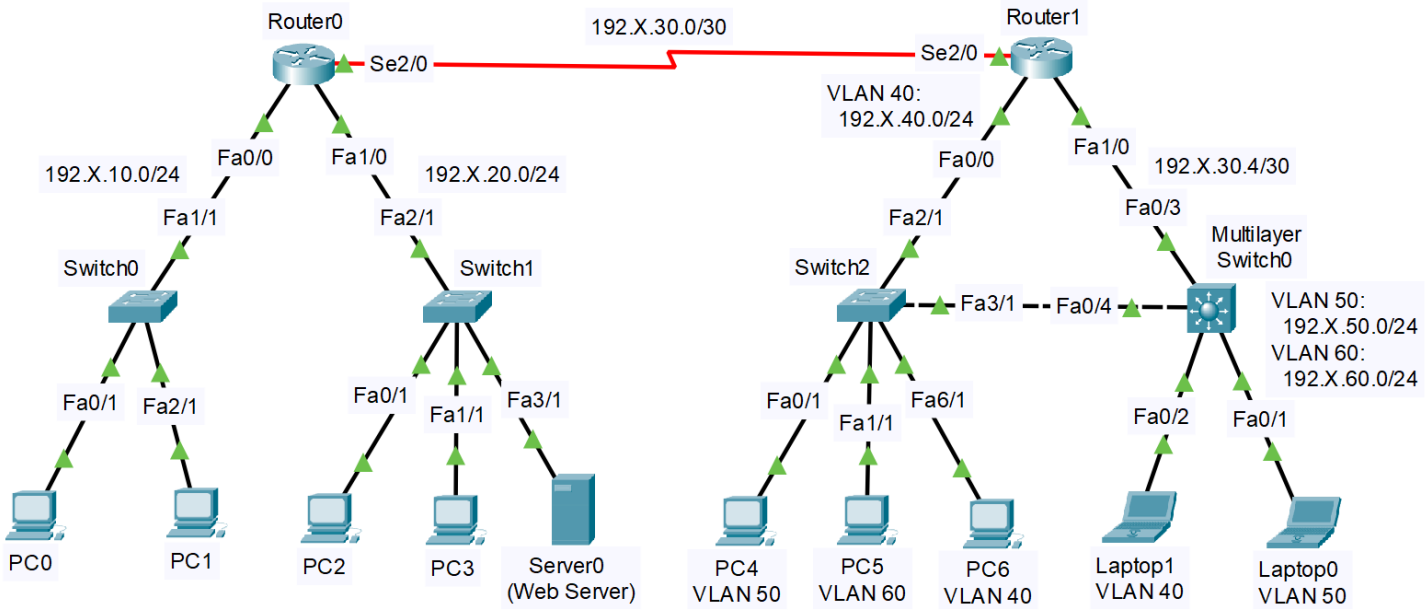

Figure 1: Access control lists topology.

Assign appropriate IP addresses to all PCs and routers, ensuring that each device receives a unique address within the valid host range of its respective network. The value of **X** in the network address corresponds to the *last two digits* of your student ID. For example, if the network address is **192.X.10.0/24** and your student ID is **1230105**, then **X = 05**, resulting in the network address **192.5.10.0/24**.

A) Configuring Router Sub-Interface

We need to create a sub-interface on **Router1** for **VLAN 40**, which will serve as the ***default gateway*** for that VLAN. The steps are as follows:

1. Enable the main interface:

```
Router(config)# interface Fa0/0
Router(config-if)# no shutdown
```

2. Create and configure the sub-interface for **VLAN 40**:

```
Router(config)# interface Fa0/0.40
Router(config-subif)# encapsulation dot1Q 40
Router(config-subif)# ip address 192.X.40.1
                      255.255.255.0
```

B) Configuring VLAN Virtual Interfaces

To enable **Multilayer Switch0** to act as the ***default gateway*** for **VLANs 50** and **60**, we need to configure the corresponding switch virtual interfaces (SVIs). To configure the SVI for **VLAN 50** with the IP address `192.X.50.1/24`, use the following commands:

```
Switch(config)# interface vlan 50
Switch(config-if)# ip address 192.X.50.1 255.255.255.0
```

You should configure **VLAN 60**'s SVI is the same manner.

C) Converting a Switch Port to a Routed Port

To assign an IP address to the port on **Multilayer Switch0** that connects to **Router1**, we must convert port `FastEthernet 0/3` from a switch port to a routed port, then assign an IP address (e.g., `192.X.30.6/30`):

```
Switch(config)# interface fa0/3
Switch(config-if)# no switchport
Switch(config-if)# ip address 192.X.30.6 255.255.255.252
```

D) Configuring OSPF Routing

We will configure the OSPF routing protocol on **Router0**, **Router1**, and **Multilayer Switch0**. Note that Layer 3 switches have routing disabled by default. To enable routing and configure OSPF on **Multilayer Switch0**:

```
Switch(config)# ip routing
Switch(config)# router ospf 1
Switch(config-router)# network 192.X.30.4 0.0.0.3 area 0
Switch(config-router)# network 192.X.50.0 0.0.0.255 area 0
Switch(config-router)# network 192.X.60.0 0.0.0.255 area 0
```

Repeat the OSPF configuration appropriately for **Router0** and **Router1**.

E) Creating VLANs

We must create all required VLANs on **Switch2** and **Multilayer Switch0** to ensure proper VLAN operations. To create **VLAN 40** on **Switch2**:

```
Switch(config)# vlan 40
Switch(config-vlan)# exit
```

Similarly, create **VLANs 50** and **60** on **Switch2**, and ensure all VLANs are created on **Multilayer Switch0**.

F) Configuring the Trunk Ports

To enable VLAN traffic between **Router1 and Switch2** and between **Multilayer Switch0 and Switch2**, configure the following ports as trunk ports:

- **Switch2:** `FastEthernet 2/1` and `FastEthernet 3/1`.
- **Multilayer Switch0:** `FastEthernet 0/4`.

To configure **Multilayer Switch0** port `FastEthernet 0/4` as a trunk port:

```
Switch(config)# interface fa0/4
Switch(config-if)# switchport trunk encapsulation dot1q
Switch(config-if)# switchport mode trunk
```

Note: When one end of a link is configured as a trunk, the other end often automatically switches to trunk mode. Therefore, you may not need to manually configure **Switch2**'s `FastEthernet 3/1` port. However, you must configure **Switch2**'s `FastEthernet 2/1` port explicitly as a trunk port.

G) Configuring Access Ports

To assign an interface to a VLAN on **Switch2** and **Multilayer Switch0**, access the relevant port and apply the access VLAN command. For example, to assign `FastEthernet 0/1` on **Multilayer Switch0** to **VLAN 50**, use the following commands:

```
Switch(config)# interface Fa0/1
Switch(config-if)# switchport access vlan 50
```

Ensure that you configure all access ports for PCs and laptops on both **Multilayer Switch0** and **Switch2** according to their assigned VLANs.

Important:

After completing the steps in this section, you should have a network with **full connectivity**. Before proceeding to any configuration tasks in the following sections, ***make sure to save the main topology*** as a reference copy.
For each task related to **Standard ACLs** and **Extended ACLs**, create a ***duplicate of the main topology*** and perform the task on that copy.
You may name each copy descriptively, such as:
Standard_ACL_Task_A, **Extended_ACL_Task_B**, and so on.

9.2.2 Configuring Standard ACLs

In this section, we will configure **standard ACLs** to regulate communication between devices in the network. Because standard ACLs do not consider the destination, they should be ***placed as close to the destination as possible*** to avoid unintentionally blocking other traffic. Standard ACLs use numbers in the range `1` to `99`.

A) Prevent PC0 from Accessing Network 192.X.20.0/24

To block traffic originating from `PC0` (e.g., `192.X.10.2`), create a standard ACL on **Router0** using either of the following methods:

Method 1:

```
Router(config)# access-list 10 deny host 192.X.10.2
Router(config)# access-list 10 permit any
```

Method 2:

```
Router(config)# access-list 10 deny 192.X.10.2 0.0.0.0
Router(config)# access-list 10 permit any
```

The `permit any` statement is necessary because all ACLs have an implicit `deny all` at the end. Next, apply the ACL to interface `FastEthernet 1/0` in the `outbound` direction:

```
Router(config)# interface fa1/0
Router(config-if)# ip access-group 10 out
```

This configuration prevents **PC0** from reaching any device in network `192.X.20.0/24`.

B) Restrict Access to Network 192.X.10.0/24 — Allow Only PC3

Create an ACL on **Router0** that permits only **PC3**'s IP address, and then apply it `outbound` on the interface leading to the `192.X.10.0/24` network (i.e., `FastEthernet 0/0`).

C) Prevent Network 192.X.10.0/24 from Accessing Network 192.X.20.0/24

To achieve this, create a standard ACL on **Router0** that block all traffic from the `192.X.10.0/24` network (The `permit any` statement is required at the end to allow all other traffic that is not explicitly denied by previous rule). Then, apply the ACL to interface `FastEthernet 1/0` in the `outbound` direction.

D) Prevent PC1 from Accessing VLAN 50 (Allow All Other Traffic)

On **Multilayer Switch0**, create a standard ACL that denies **PC1**'s IP (e.g., `192.X.10.3`) and permits all other traffic. Apply the ACL `outbound` on the SVI for **VLAN 50**.

E) Restrict Access to VLAN 40 — Allow Only VLAN 60

Create a standard ACL on **Router1** that allows only traffic from the `192.X.60.0/24` network. Then, apply it `outbound` on the sub-interface for **VLAN 40** (i.e., `Fa0/0.40`). This configuration ensures that only **VLAN 60** can reach **VLAN 40**, while all other traffic is blocked by the default implicit deny.

9.2.3 Configuring Extended ACLs

In this section, we will configure **extended ACLs** to control specific types of traffic between hosts. Since extended ACLs filter by both source and destination, they should be ***placed as close to the source as possible*** to prevent unnecessary traffic from traversing the network. Extended ACLs use numbers in the range `100` to `199`.

A) Prevent PC0 from Accessing PC2 (Allow All Other Traffic)

To block IP communication from **PC0** (e.g., `192.X.10.2`) to **PC2** (e.g., `192.X.20.2`), create an extended ACL on **Router0** using one of the following methods:

Method 1:

```
Router(config)# access-list 101 deny ip host 192.X.10.2 host
                192.X.20.2
Router(config)# access-list 101 permit ip any any
```

Method 2:

```
Router(config)# access-list 101 deny ip 192.X.10.2 0.0.0.0
                192.X.20.2 0.0.0.0
Router(config)# access-list 101 permit ip any any
```

Then, apply the ACL to interface `FastEthernet 0/0` in the `inbound` direction:

```
Router(config)# interface fa0/0
Router(config-if)# ip access-group 101 in
```

B) Limit VLAN 40 Access: Allow Only PC6 to Reach PC1

To allow only **PC6** (e.g., `192.X.40.2`) to access **PC1** (e.g., `192.X.10.3`), create an extended ACL on **Router1** that permits this specific traffic and blocks all other traffic originating from **VLAN 40**. Since there is an implicit deny all at the end of every ACL, no additional deny statement is required. Apply the ACL `inbound` on the interface receiving traffic from **VLAN 40** (i.e., `Fa0/0.40`).

C) Block HTTP Requests from PC0 to Server0

First, ensure the HTTP service is enabled on **Server0** (e.g., `192.X.20.4`). Then, configure an extended ACL on **Router0** to block only HTTP traffic (TCP port 80) originating from **PC0** (e.g., `192.X.10.2`) to **Server0**. Other types of communication (such as ICMP/ping) should remain unaffected. Be sure to include the `permit ip any any` statement in the ACL to allow all other traffic. Finally, apply the ACL `inbound` on the router interface closest to **PC0**.

D) Restrict Access to PC3 from VLAN 50 — Allow Only Laptop0

Create an extended ACL on **Multilayer Switch0** that permits traffic destined for **PC3** (e.g., `192.X.30.3`) only if it originates from **Laptop0** (e.g., `192.X.50.3`). Apply the ACL `intbound` on the SVI for **VLAN 50**.

E) Restrict Telnet Access to Router1's Serial 2/0 — Allow Only PC0

Only **PC0** (e.g., `192.X.10.2`) should be allowed to access **Router1**'s `Serial 2/0` interface (e.g., `192.X.30.2`) via Telnet. All other hosts must be denied Telnet access to this interface.

1. **Enable Telnet on Router1**

   Configure the VTY lines on **Router1** to allow Telnet access with authentication:

```
Router(config)# line vty 0 4
Router(config-line)# password <student-name>
Router(config-line)# login
Router(config-line)# exit
Router(config)# enable password <student-ID>
```

   After configuring **Router1** as described above, **PC0** will be able to initiate a Telnet connection to **Router1**'s `Serial 2/0` interface, as illustrated in Figure 2.

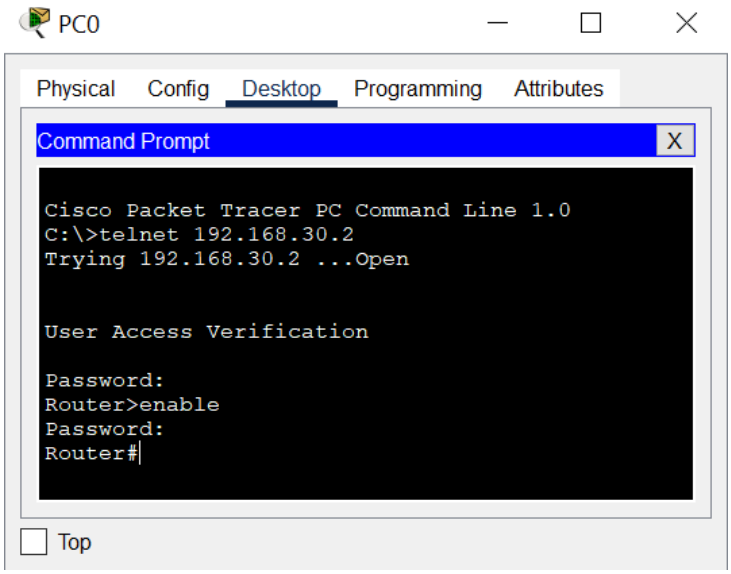

Figure 2: Successful Telnet connection from PC0 to Router1's Serial 2/0 interface.

2. **Permit Telnet Access from PC0 Only (Configured on Router0)**

   Create an extended ACL on **Router0** to allow Telnet (TCP port 23) access to **Router1**'s `Serial 2/0` only from **PC0**, and block all other Telnet traffic to that interface. Other types of traffic should still be allowed.

```
Router(config)# access-list 105 permit tcp host
                192.X.10.2 host 192.X.30.2 eq telnet
Router(config)# access-list 105 deny tcp any host
                192.X.30.2 eq telnet
Router(config)# access-list 105 permit ip any any
```

   Figure 3 shows the created extended ACL on **Router0**. The number of matches shown in an access list entry—such as `16 match(es)` or `24 match(es)`—indicates

how many packets have matched (i.e., triggered) that specific ACL rule since the ACL was applied or last cleared.

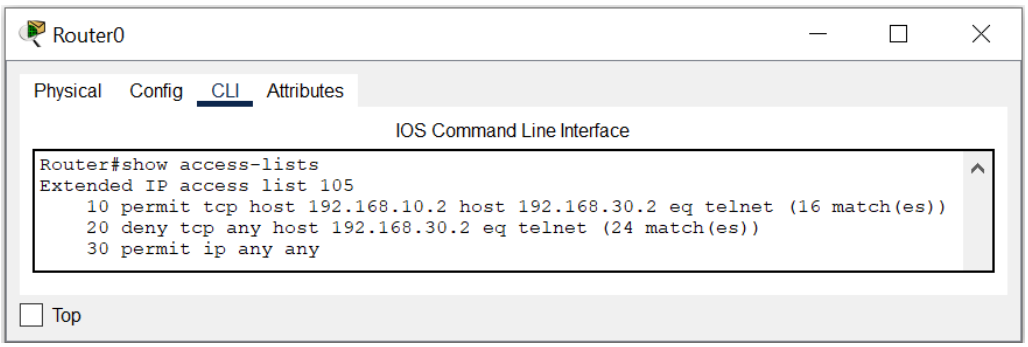

Figure 3: Extended ACL configuration on Router0.

Apply the ACL `inbound` on interfaces `FastEthernet 0/0` and `FastEthernet 1/0`:

```
Router(config)# interface fa0/0
Router(config-if)# ip access-group 105 in
Router(config-if)# interface fa1/0
Router(config-if)# ip access-group 105 in
```

3. **Block Telnet from VLAN 40 (Configured on Router1)**

   Create an extended ACL on **Router1** to deny Telnet traffic from **VLAN 40** to its `Serial 2/0` interface. Then, apply the ACL `inbound` on the sub-interface for **VLAN 40** (i.e., `Fa0/0.40`).

4. **Block Telnet from VLANs 50 and 60 (Configured on Multilayer Switch0)**

   Create an extended ACL on **Multilayer Switch0** to block Telnet access from **VLANs 50** and **60** to **Router1**'s `Serial 2/0`. Then, apply the ACL `intbound` on the SVIs for **VLANs 50** and **60**.

9.3 ToDo

This section will be provided by the instructor.