

350-018 Lab

[Access Lists](#)
[Basic Router Operation](#)
[Firewalls](#)
[ISDN](#)
[ISDN and DDR](#)
[ISIS](#)
[L3 VPNs](#)
[NAT](#)
[Network Security](#)
[Other VPNs](#)
[Securing Communications I](#)
[Securing Communications II](#)
[Security](#)

Access Lists Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Access Lists: Tricks of the Trade

Access lists are a general Cisco mechanism that allows you to be selective in the traffic you forward -- more selective than routing alone. They operate on individual packets, which distinguishes them from the new techniques of traffic engineering. Access lists (and their more powerful cousins, such as route maps) are designed to let you specify selective handling for certain traffic, beyond the rules established by traditional destination-based forwarding.

350-018 Labs

Access Lists

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Access Lists Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Access Lists: Tricks of the Trade

[Introduction](#)

[Access Control Lists and Access Lists](#)

[Pattern Matching](#)

[Actions](#)

[Permit and Deny](#)

[A First Performance Consideration](#)

[Call Setup](#)

[Quality of Service](#)

[Installing an Access List](#)

[Managing Access List Configuration](#)

[Modifying and Deleting Access Lists: Just Say "No!"](#)

[Documenting Access Lists](#)

[Applying Access Lists](#)

[Interfaces](#)

[Virtual Terminals](#)

[Routing Processes](#)

[Access List Processing](#)

[How to Get in Trouble](#)

[Security](#)

[Defining Security Classes of Users](#)

[Specifying Access Control](#)

[Planning Access List Placement](#)

[Tip](#)

[IP Access List Considerations](#)

[IP Access List Performance](#)

[Ingress Filtering](#)

[IP Extended Access Lists](#)

[IP Protocol Type](#)

[Leaking Routing Updates](#)

[ICMP Specific](#)

[IGMP Specific](#)

[TCP and UDP Specific](#)

[Tip](#)

[Advanced Mechanisms](#)

[Reflexive IP Access Lists](#)

[Dynamic Access Lists](#)

[Time Ranges in Extended Access Lists](#)

[Maps](#)

[Working with Access Lists](#)

[Tools for Troubleshooting Access Lists](#)

[Context-Sensitive Help is your Friend](#)

[Logging with Access Lists](#)

[Conclusion](#)

[References](#)

Introduction

A router's job is to determine which interface the packet is to be passed through based on the destination address included in the packet header. This is a simple concept based on the premise that the router should forward all packets received to the network that contains the destination node.

A challenge to real-world routers, however, is often not how fast they can forward packets, but how quickly they can decide which packets should *not* be forwarded.

Now, what if we didn't want to transfer every packet that requested access through to any segment on the network and ultimately to any node? What if we were to say that we didn't want to accept packets from certain sources or that we didn't want certain nodes or networks to communicate with

other nodes or networks? How would we tell the router not to pass along every packet that it receives, but instead to check the packet against a list we provided? Cisco provided answers to these questions with access control lists usually referred to as simply "access lists."

Access Control Lists and Access Lists

It's worth drawing a distinction between *access list* and *access control list*. Access lists are a general Cisco mechanism that allows you to be selective in the traffic you forward -- more selective than routing alone. They operate on individual packets, which distinguishes them from the new techniques of traffic engineering. Conceptually, traffic engineering creates alternate routes rather than creates per-packet routes, although the implementation blurs the two ideas. Nevertheless, traffic engineering is outside the scope of this discussion.

Access **control** lists are specifically intended for security. There are access control lists on many types of networked devices, not just routers. **tcpwrapper**, for example, is a public domain tool for UNIX hosts.

Access lists (and their more powerful cousins, such as *route maps*) are designed to let you specify selective handling for certain traffic, beyond the rules established by traditional destination-based forwarding. Applications for access lists and related functions include security, performance optimization, triggering events such as dialing, etc. The focus of this paper is on Cisco access lists, not the broader problem of access control.

They can be configured to filter packet traffic for all routed network protocols. The concept is simple: A list is created to tell which packets are permitted and which are denied. When a packet is received, it's compared to the list and is either accepted and passed along or rejected and dropped based upon permit or deny permissions that have been stated in the list.

Different types of access lists use different criteria to determine which packets are routed and which are not. Standard access lists use the bare minimum of criteria, usually the source address. Extended access lists can use a variety of additional criteria including destination address, protocols, and ports.

Pattern Matching

The first part of filtering involves defining the traffic that should require special handling. This is an application of pattern matching. Especially when making decisions based on addresses, wildcard masks commonly are used to let one rule match multiple addresses.

Other kinds of rules can specify things such as ranges of TCP or UDP port numbers. See Figure 1 for some of the things that access lists and route maps can look for.

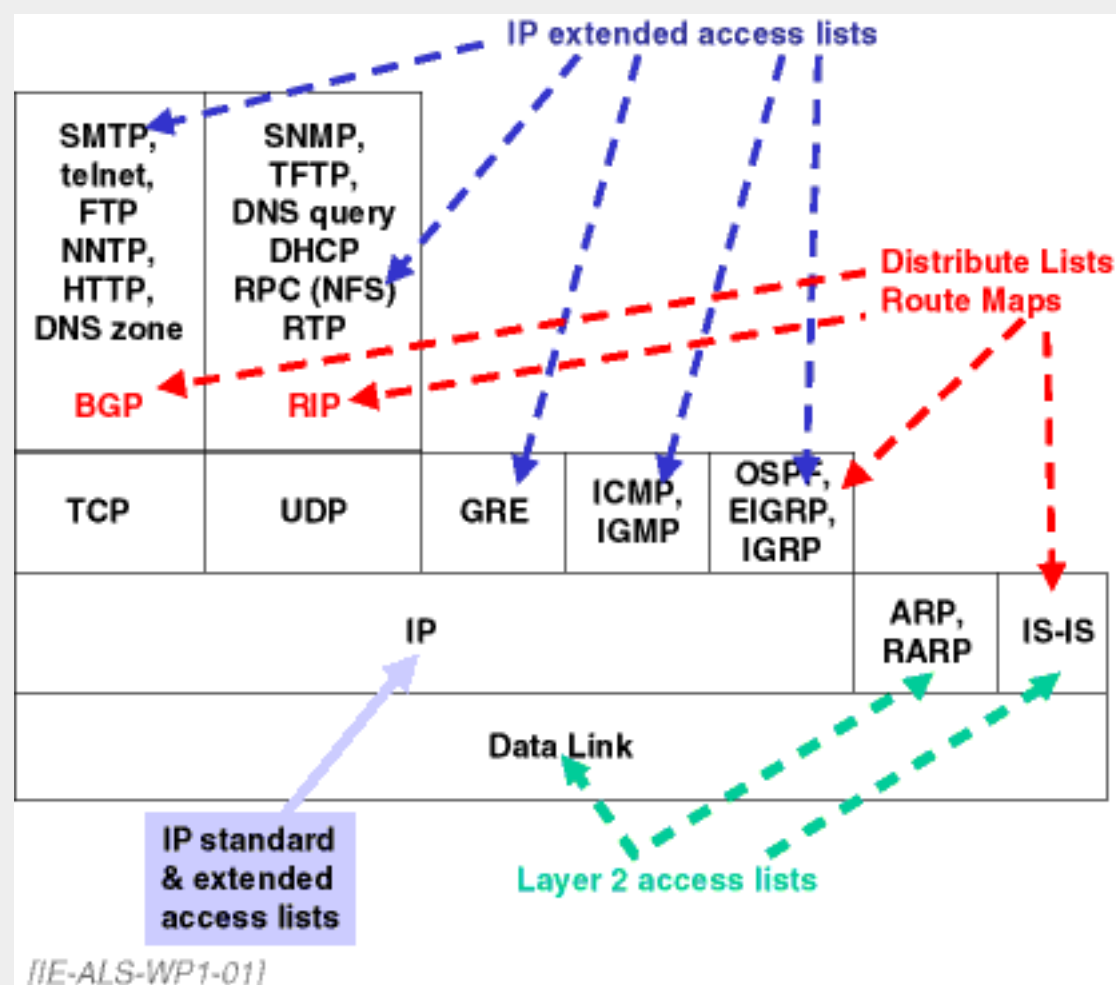


Figure 1.

Actions

When dealing with routed traffic, the main actions performed by access lists are permitting or denying packets access through the interface.

Specialized access lists can alter fields in those packets, such as the IP precedence field used for quality of service signaling. Variant access lists can do other QoS-related functions such as assigning packets to outbound queues.

When using distribute lists to deal with routing packets, the access lists control what routing information goes in or out of a routing process. Distribute lists either permit or deny flow, as opposed to the more powerful route maps, which also allow routing packets to be modified.

Permit and Deny

When a packet reaches an interface, the interface can either permit the packet to pass or deny it entrance. The interface can't perform any other function on that packet.

The best analogy I can give would be to compare this to a popular nightclub. This club has multiple entrances, and at each entrance is a doorman. In his hand is a clipboard with a list of names. When you approach the doorman and he identifies you, he then looks at his list, starting from the top, and compares your name against it, attempting to find a match. If he locates your name and the list specifies that you are permitted to enter, he allows you to pass through the entrance. If the list states that you are to be denied entry, you're turned away. If he goes through the entire list without finding a match, the default is to deny you entrance. (Keep the top-down sequence in mind when designing access lists.)

The nightclub analogy is based upon the control of the entrances to the club. With access lists, not only can you control which packets are allowed in, but you can also control which packets are allowed out. Let's assume the nightclub has a VIP entrance/exit. At this VIP entrance are two lists: one for those who are trying to enter, as we have already described, and another to filter those who are attempting to leave through that door. The second list would be applied in the same manner as the first, from the top down, and it would only be able to either permit or deny. Of course, if no permit were issued by the time the end of the list was reached, the result would be that the implicit "deny all" would deny passage.

Now, let's consider a couple of points about this analogy:

- Your name is compared against those on the list from the top down, until a match is made. Then the permit-or-deny condition is determined. After a match is made, any other information further down the list is never accessed. Keep this in mind, because this makes the order of the entries in your access list very important.
- If there is no match by the time the entire list has been checked, then the default is to deny all. This behavior is referred to as the *implicit deny all*. If you haven't provided permission to pass through somewhere in the access list, you need to include a **permit any** statement at the end or all traffic not explicitly defined will be discarded.
- All access lists need at least one "permit" statement. If your list consists entirely of "deny" statements, then you have effectively "shut down" the interface -- no packets will be allowed to pass.

A First Performance Consideration

Now, the best way to keep this line moving quickly is to place the classifications that affect a majority at the top. If you have access list statements that affect large groups, keep them close to the top of the access list to improve throughput. For example: If all ladies are to be permitted then make this the first line so the bouncer doesn't take time looking at each name before finding "permit all ladies" at the bottom.

Call Setup

Dial-on-demand services use both access lists and dialer lists. A dialer list allows you to do a logical OR of access lists of different types. For example, a dialer list could include an IP access list and an IPX access list.

Quality of Service

Specialized access lists allow various aspects of quality of service to be specified. The access list variously can describe the traffic to which the QoS policy is to be applied (e.g., in generic traffic shaping), or can describe the QoS policy itself (e.g., in rate limiting).

Installing an Access List

The basic structure of access lists is presented in the CCNA Tutorial on Security.

There are two steps involved in the implementation of access control lists:

- Configuring the access list
- Applying the access list to an interface or process. There can be nuances to applying specific types of access lists; see the section "[Applying Access Lists](#)" below.

First, you need to create the access list. Then, you need to determine which interface(s) you wish to apply it to and whether it will be applied to all incoming or outgoing packets. You can create multiple access lists on a router. You can also apply the same access list to multiple interfaces. Remember, though, you can only apply one access list in each direction per interface.

There are many different types of access lists, based on and classified by the protocol that they support. Traditionally, access lists have been identified by numbers. The numeric range of the number, shown in Table 1, also identifies the protocol family to which the list applies.

Table 1. Access List Numbers

| Numeric Range | Protocol Family |
|---------------|------------------------------------|
| 1-99 | IP standard access list |
| 100-199 | IP extended access list |
| 200-299 | Protocol type-code access list [2] |
| 300-399 | DECnet access list |
| 600-699 | AppleTalk access list [1] |

| | |
|-----------|---|
| 700-799 | 48-bit MAC address access list [2] |
| 800-899 | IPX standard access list |
| 900-999 | IPX extended access list |
| 1000-1099 | IPX SAP access list |
| 1100-1199 | Extended 48-bit MAC address access list [2] |

1: see Priscilla Oppenheimer's CCIE AppleTalk Tutorial

2: see Leigh Anne Chisholm's CCNA Switching Tutorial

Unfortunately, the use of the number as identifying the pattern to match (i.e., the global command) is inconsistent with the usage that applies access lists to interfaces. An extended IP access list, for example, is identified with a number of 100 to 199, but the **access-group** command that invokes this list on an interface has the format **ip access-group number**. Just consider this inconsistency one of those that you need to live with in order for a relationship to work, just like beard residue in the sink or pantyhose over the shower curtain. In this case, your relationship is with IOS.

A recent addition to the IOS is the ability to use a name rather than a number for certain access lists. Although this ability certainly improves readability, the reason it was added is that large ISPs needed more than 100 access lists of a given type on a single router. Named access lists removes the numbering range restriction. Using named access lists was introduced in IOS 11.2, and is available only for IP packet and route filters.

Table 1 shows the major classes of access lists and the numbering ranges they use. Each protocol class has a list ranging from x00 through x99. The main protocols we'll discuss in this paper are in the first group. The additional classes are seldom used and will not be covered. In this paper we will pay particular attention to IP and IPX access lists since they are the ones you will be using most often. However, for certification exams, you will be expected to know all the different access list types and how they are applied.

The IOS knows the numbering schemes for the different types of access lists. This is why it is important to use the correct numbering when you create an access list. When in doubt, use the **access-list ?** help command.

Once we have determined the purpose of our access list and whom we will restrict, we then know the numeric range to use for our list. Next we need to properly configure our list.

Some access lists have names rather than numbers. Other IOS commands, which are based on access lists, typically start numbering with 1. Examples of these other commands include **dialer-list**, **priority-list**, and **queue-list**.

Managing Access List Configuration

Once you've created an access list, it remains in the router's configuration until it's deleted. It's available for use on any interface on the device, but has no effect on any interface unless it's applied. To apply the access list to an interface, you need to be in configuration mode. Then, you specify the interface to which you're applying it and add the access list number. Access lists are applied as access-groups in the following format:

```
protocol access-group access-list_number {in | out}
```

Here's an example:

```
interface ethernet0
ip access-group 7 in
```

This example would apply standard IP access list 7 to interface Ethernet 0 against all inbound traffic. Remember, you don't add with an **ip access-list** command. Instead, it's referred to with **ip access-group**. You then specify it for traffic going in or out of the interface. You can have only one inbound access list per interface, but you can also include an outbound access list.

Here, we apply extended access list 121 to all traffic outbound through interface Ethernet 0:

```
interface ethernet0
ip access-group 121 out
```

Although you're limited to one access list inbound and one access list outbound per interface, the two lists can be of different types. For example, the inbound filter might be a Standard IP access list, while the outbound filter could be an Extended IP access list.

Modifying and Deleting Access Lists: Just Say "No!"

The rule for modifying lines of an access list, once entered, is simple. You can't. You can append lines to the end, but if you want to modify or delete individual lines, you need to delete and reinstall the entire list.

Your device may have a number of different access lists, particularly switches that have dozens or even hundreds of interfaces. These access lists need to be controlled. You need to apply them to the interfaces you want them to affect or delete them from those interfaces. Sometimes you also need to remove them entirely from the active configuration. In this section, we'll take a look at how to manipulate the access lists you've configured.

In all but the simplest environments, it is worth maintaining access lists as distinct files on a server. There are several Cisco mechanisms that let you add or delete parts of a configuration file.

Sometimes you'll make changes that disable filtering on certain interfaces. At some point, you may no longer need them and wish to delete them. The commands to delete them from your database are simple. Just use the command word "no" and specify the minimum number of keywords and arguments needed to delete the proper access list.

To disable use of a specific access list from a specific interface, use the following command format:

```
no {ip | ipx} access-group access-list_number {in | out}
```

For example:

```
interface ethernet0
no ip access-group 121 out
```

If you want to delete the access list itself, use the following command:

```
no access-list access-list_number
```

For example:

```
no access-list 121
```

To delete the access list for a specific protocol, use the following command:

```
no access-list access-list_number {permit | deny} protocol
```

For example:

```
no access-list 201 permit tcp
```

When deleting access lists, be sure they are no longer in use. If an access list is in effect on an interface, and there is no corresponding access list number, then no filtering is in effect. The access list is useless and all inbound and outbound traffic is permitted to pass through the interface.

Documenting Access Lists

You have always had the ability to include comments in Cisco configuration files, which certainly can help you understand the purpose of access lists. Unfortunately, comments are not preserved in NVRAM or network loaded files.

In well-run networks, access lists of any complexity should be maintained on configuration servers and then downloaded. Comments, therefore, should be available. Nevertheless, there are situations where the master files are not available. As with the **description** subcommand of **interface**, there is now a way to store comments in loadable configuration files.

Cisco IOS 12.0 added a new feature to access lists. Now, you're able to write a comment (remark) for an entry in a numbered access list. This remark can be up to 100 characters long. This is an excellent tool to explain the purpose of the access list, along with who created it and when. The format for remarks is

```
access-list access-list_number remark remark
```

Here is an example:

```
access-list 233 remark See Policy Guide
    dated 05/30/00 pg 14 ref Accounting
    Department restrictions
```

To remove the remark, use the **no** form of this command:

```
no access-list access-list_number remark
```

For example:

```
no access-list 233 remark
```

Applying Access Lists

It's excellent practice to start the configuration of every access list with a "no" statement that clears the list:

```
no access-list 1
! comments on access-list
access-list 1 ! first statement of list
```

This prevents confusion caused by inadvertent merging of existing list statements with statements loaded from a server or NVRAM.

You need to apply access lists to specific interfaces and processes. For any specific application of an access list, you can place up to two access lists: one inbound and one outbound. There can't be more than one list per direction per interface.

Interfaces

You apply access lists to interfaces with an **access-group** command. You may have only one access group of each protocol type in each direction.

For example, you could have input access lists for IP and IPX, an output list for IP, and an output list for AppleTalk. You could not have two output access lists for IP.

Virtual Terminals

The **access-class** command applies an IP access list to a virtual terminal line.

Routing Processes

distribute-list commands apply access lists to routing processes. While the access lists invoked by **distribute-list** look like any other IP access lists, the IP address in them has a meaning different from that of the address in a list invoked by an **access-group**.

In a **distribute-list**, the IP address that is matched is the address **carried by** a routing update, **not** the source address of the packet carrying the routing update.

Access List Processing

After receiving a packet at the inbound interface, the software checks the packet against the access list. If the access list permits the packet, the software continues to process it. If the access list rejects the packet, then the software discards it and returns an Internet Control Message Protocol (ICMP) Destination Unreachable message with a cause code of "Communication Administratively Prohibited."

To create effective access lists, you need to understand how they are utilized by the interface of the device. Here are the steps the router takes when applying an access list.

1. The packet is compared against the access list in sequential order.
2. It's compared line by line from the top down until a match is made (note the importance of order).
3. As soon as a match is made, the action specified on that line is taken. The rest of the access list is skipped.
4. An implicit "deny all" statement exists at the end of each list. This means that if none of the previous statements has matched the packet, then it will be discarded when it reaches the end of the list.

Cisco documentation refers to "Destination Unreachable" as "Host Unreachable." The RFCs refer to this as "Destination Unreachable."

How to Get in Trouble

When we first explained what an access list is, we discussed how the list is accessed in order until a match is made. This order is very important because once an action is taken upon a packet, the packet is not processed further. Only one list is used per direction per interface, and only one decision is made for each packet.

Let's see how the order of the entries would affect our previous example. Suppose that, instead of your list being configured like this:

```
access-list 6 permit 140.88.0.1
access-list 6 deny 140.88.0.0 0.0.255.255
access-list 6 permit 140.0.0.0 0.255.255.255
```

it is configured like this:

```
access-list 2 permit 140.88.0.1
access-list 2 permit 140.0.0.0 0.255.255.255
access-list 2 deny 140.88.0.0 0.0.255.255
```

Now a packet is transmitted with node address 140.88.67.44. It reaches the interface that has access list 6. When the packet reaches line 2 -- the statement **deny 140.88.0.0 0.0.255.255** -- it will match, and the **deny** will cause it to be discarded. If that same packet reaches an interface with access list 2 applied, it will reach line 2, and the statement **permit 140.0.0.0 0.255.255.255** will cause it to be permitted, because it matches that criteria. The fact that line 3 would deny the packet is irrelevant because as soon as a match is made, the packet's fate is determined and the rest of the list is not checked.

Security

Access lists commonly are used in security, but that is by no means their only application. Security, in turn, has many other features besides access control.

Before implementing any security solution, you really need to have a security policy. In practice, this is a short document (not more than 1-2 pages; if

longer, people won't read it) approved by top management.

The core of a security policy is a *trust model*, which specifies which users (called subjects in formal security policy) can perform what, if any, operations on servers (called objects). Other parts of the security policy are administrative, specifying such things as actions to be taken if there is a violation, and who has the authority to grant security privileges.

Assume your organization has four sites: Tulsa, Dallas, Phoenix, and Las Vegas. All Internet access goes through Dallas, which has an address-translating firewall separate from the main corporate router. There is also a public server LAN in Dallas.

You use private address space internally and have a small amount of registered address space in the DMZ.

```
Dallas block: 172.16.0.0/16
Tulsa block: 172.17.0.0/16
Las Vegas block: 172.18.0.0/16
Phoenix block: 172.19.0.0/16
Core block: 192.168.0.0/24
Public DMZ: 192.0.2.0/28
```

Users and servers at each of these sites connect to VLAN switches, and each user belongs to a departmental VLAN. Servers at a given site also have their own VLANs. The servers are on separate VLANs to keep backup traffic off the user VLAN.

Each location's set of site servers can communicate with the site servers at other sites. Email, for example, is distributed across these servers. Users at one site, however, are not allowed to directly access the site server at a different site.

| | | |
|-----------|------------------------|---------------|
| Dallas | Engineering Users | 172.16.0.0/24 |
| | Shipping Users | 172.16.1.0/24 |
| | Systems Administration | 172.16.3.0/24 |
| | Engineering Servers | 172.16.4.0/24 |
| | Site Servers | 172.16.5.0/24 |
| | Perimeter Network | 172.16.6.0/24 |
| Tulsa | Engineering Users | 172.17.0.0/24 |
| | Shipping Users | 172.17.1.0/24 |
| | Shipping Servers | 172.17.3.0/24 |
| | Site Servers | 172.17.4.0/24 |
| Las Vegas | Engineering Users | 172.18.0.0/24 |
| | Shipping Users | 172.18.1.0/24 |
| | Accounting Users | 172.18.2.0/24 |
| | Accounting Servers | 172.18.3.0/24 |
| | Site Servers | 172.18.4.0/24 |
| | Shipping Users | 172.18.1.0/24 |
| Phoenix | Site Servers | 172.19.1.0/24 |
| | Shipping Users | 172.19.2.0/24 |

Let's assume you wanted all the users in the accounting department to be able to access all the servers in the accounting network, but you wanted to restrict your shipping and engineering staffs from doing so. You do, however, want all employees to be able to access the time & attendance reporting application that runs on TCP port 2200.

Dallas, Tulsa, and Las Vegas all have some type of corporate-wide server: engineering in Dallas, shipping in Tulsa, and accounting in Las Vegas. Yes, you may raise an eyebrow at having accounting in Las Vegas.

Each site also has site-level servers for backup, printing, etc. that should only be accessible to that site and to the system administrator in Dallas.

Defining Security Classes of Users

One of the first things to do is to identify the classes of users. If these can be identified by an address, such as a LAN workstation address, access lists can have a significant role in security. This is equally true if addresses are dynamically assigned by a process that involves authentication, such as CHAP (Challenge Handshake Authentication Protocol) on dialup lines.

Table 2. Defining Membership in Subject Classes

| Subject | Membership |
|----------------------|---|
| General Employee | 172.16.0.0/16, 172.17.0.0/16, 172.18.0.0/16, 172.19.0.0/16 |
| Dallas Employee | 172.16.0.0/16 |
| Tulsa Employee | 172.17.0.0/16 |
| Phoenix Employee | 172.19.0.0/16 |
| Las Vegas Employee | 172.18.0.0/16 |
| Engineering User | 172.16.0.4.0/24, 172.18.0.0/24 |
| System Administrator | 172.16.4.0/24 |
| Shipping User | 172.16.1.0/24, 172.17.1.0/24, 172.18.1.0/24, 172.19.1.0/24 |
| Accounting User | 172.18.2.0/24 |
| General Public | All valid addresses EXCEPT 172.16.0.0/16, 172.17.0.0/16, 172.18.0.0/16, 172.19.0.0/16 |

Once the users are defined, what can they do?

Specifying Access Control

You have only solved part of the problem when you define user classes. You also need to define what each class is, or is not, allowed to do. It's usually convenient to set up a matrix to describe the relationships of users and services, or, as services are called in formal security literature, objects.

The matrix in Table 3 is intended to show policies, not to have a cell for every access list rule. If there is no entry in a cell, access is not permitted.

Table 3. Subject and Object Relationships

| Subject | Object/Service | | | | | |
|----------------------|----------------|-------------------|------|----------|------------|-------------------------------------|
| | Site Server | Perimeter Network | Engr | Shipping | Accounting | Time & Attendance (172.18.3.3:2200) |
| Dallas Employee | Dallas | Yes | | | | Yes |
| Tulsa Employee | Tulsa | Yes | | | | Yes |
| Phoenix Employee | Phoenix | | | | | Yes |
| Las Vegas Employee | Las Vegas | Yes | | | | Yes |
| Engineering User | | Yes | | | | Yes |
| System Administrator | Yes | Yes | Yes | Yes | Yes | Yes |
| Shipping User | | | | Yes | | Yes |
| Accounting User | | | | | Yes | Yes |
| Site Server | Yes | | | | | |
| General Public | | | | | | |

Planning Access List Placement

When planning access lists, I find that the traditional way of drawing a network (Figure 2), emphasizing physical interfaces and media, is not the best way to plan where to place access lists. Never forget that a given access list works only in one direction. To control the same type of traffic in both directions, you need two access lists.

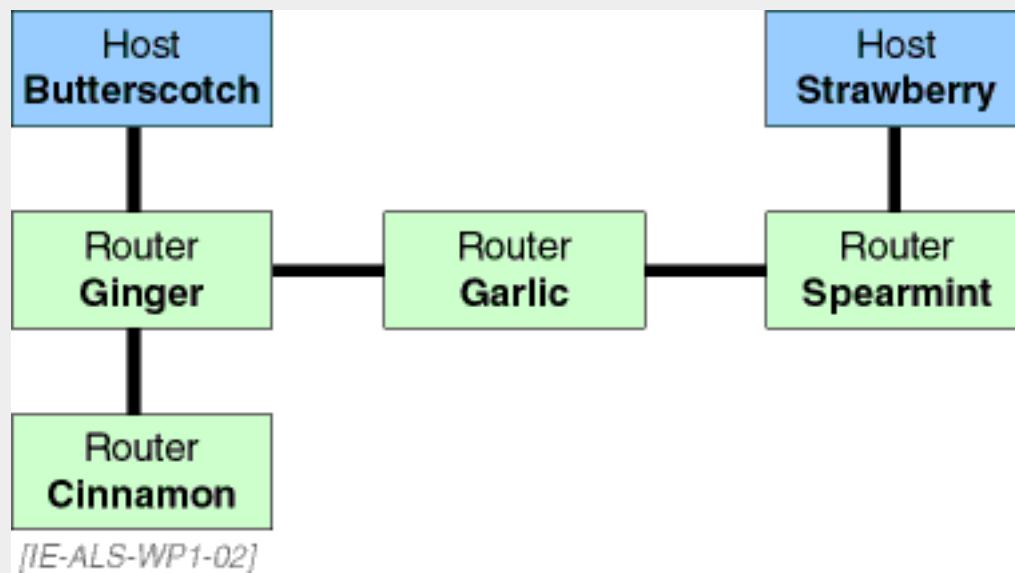


Figure 2.

My preference for access list planning is to draw two lines between each pair of devices, one in each direction. Color-coding, as shown in Figure 3, helps make the flow more clear.

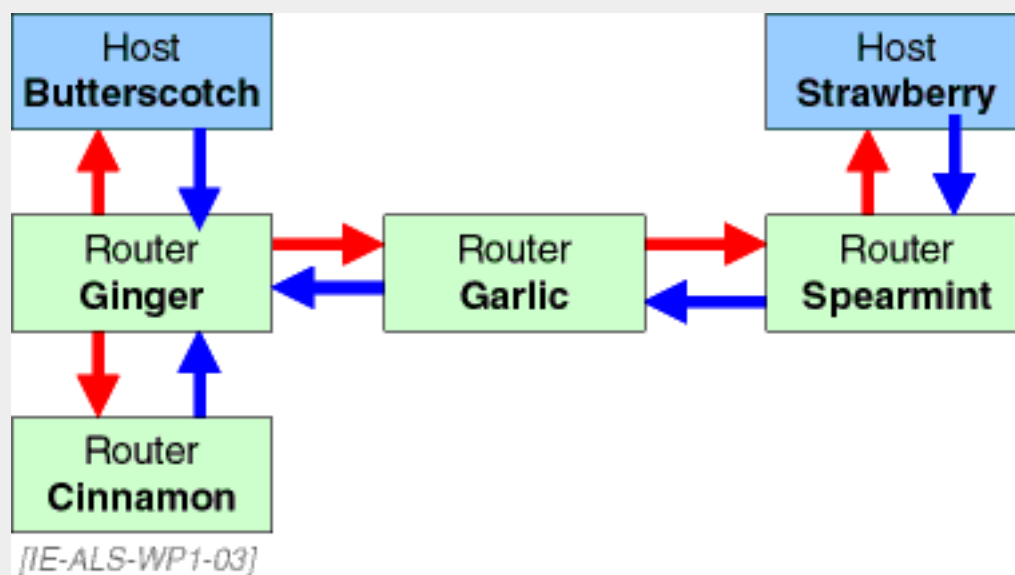


Figure 3.

Using this technique, you can show inbound access lists at the point of arrows, and outbound interfaces at the base of arrows. In Figure 4, router **garlic** has an inbound access list 102 and an outbound list 111. Router **ginger** also uses list 102 on input.

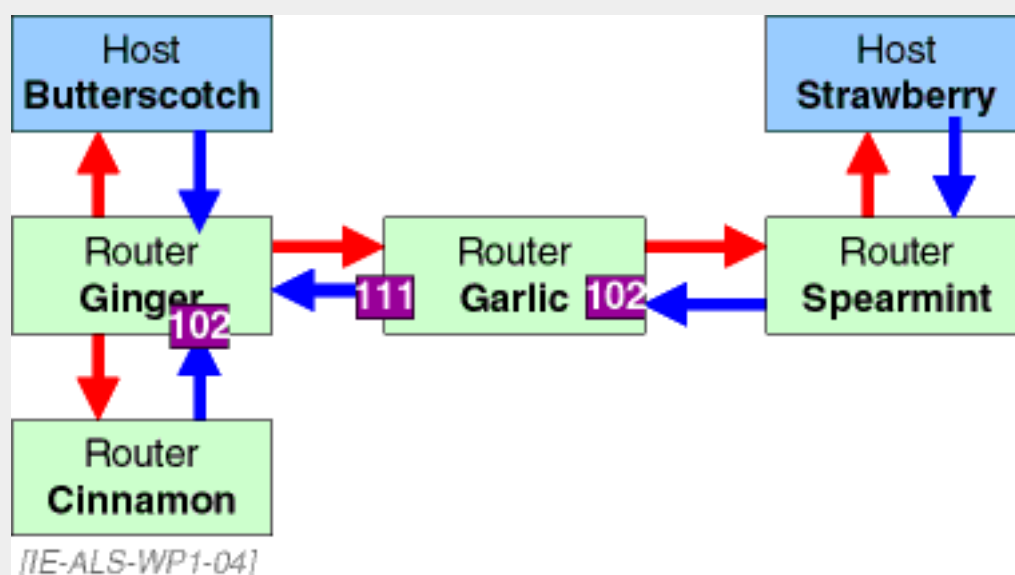


Figure 4.

IP Access List Considerations

There are two basic kinds of IP access lists, with a number of advanced types that build on the basic types. Standard access lists operate on the source address only, while extended access lists operate on source and destination addresses. Extended access lists can also consider additional parameters, such as source and destination TCP/UDP port numbers or IP precedence fields.

Some of the "advanced" types are not so much completely new access list statements as additional parameters for standard and extended access lists.

IP Access List Performance

Cisco routers have a great number of switching paths. Certain access lists can force your packets into a slower switching path. Unfortunately, which

Tip

Try to make your access list numbers global for your network, so that list 102 has the same meaning in every router. This will also simplify your administration of lists on servers.

If this practice threatens to give you over 100 lists, use named access lists.

On the CCIE lab, however, you have a small enough number of routers that you can safely number your access lists uniquely.

access lists are associated with which path will depend on hardware platform and IOS version. You can tell which path is being used, however, with a **show ip interface** command.

Unless you have a router with many serial interfaces, access list performance tends not to be an issue at T1/E1 speeds and below. It can be quite significant at LAN speeds.

When configuring an access list, you should strive to provide the quickest decisions for the largest number of packets passing through the interface. You can manipulate this decision speed by organizing your list so that the lines closest to the top affect the largest number of packets.

Let's assume you want to allow access to a network of 1,000 while denying access to about 30 specific remote nodes. One way would be to configure the 30 different **deny** statements and then include a **permit any** at the end. Of course, this setup would mean that every packet from these 1,000 workstations would need to be processed through 30 statements before being permitted. Another (better) way would be to permit this entire network on the first line and then list your **deny** statements. The majority of packets would receive instant permission to pass, and all others would need to be compared against the **deny** statement. This arrangement would affect the speed by which this interface would process.

Network location is also a consideration when placing access lists. In general, standard lists should be placed close to the destination, and extended lists should be placed close to the source. This placement will allow packets to be filtered as quickly as possible without consuming bandwidth unnecessarily. You can consider this rule to be very general. Sometimes one list at the interface to the destination can be used instead of 20 lists scattered near all the restricted sources.

Table 4. Filter Placement

| Design goal | Lists primarily located on router at tier: | | | |
|----------------------------|--|---------------------------------|---|--------------------|
| | Client Access Tier | Distribution Tier | Core Tier | Server Access Tier |
| Router cost Minimization | Small routers can be cheap, but significant processing can require more expensive models | Usually best economies of scale | Large core routers should do minimum or no filtering. Your spending should be on forwarding, not filtering capabilities | |
| Bandwidth Minimization | Good | Fair | Poor | Good |
| Maximizing Maintainability | Poor | Good | Excellent | Poor |
| Maximizing Scalability | Good | Excellent | Poor | Good |

Standard IP access lists are simple to configure and can provide a majority of your access list needs. In cases that become more complex, and when you have to establish filters based on criteria other than the source address, then you have another tool: the IP extended access list.

Ingress Filtering

Malicious hacking, not to steal information but to deny service, is an ever-increasing problem in the Internet. One hacker technique is to break into one machine, and then use it as a launching pad for attacks. The packets generated by the attacking machine have random source addresses to prevent tracing back the attack.

One method to help control this problem is to take away this means of anonymity. RFC 2827 specifies the Best Current Practice for the Internet, in which service providers will accept only those customer-generated packets with a source address legitimately associated with that customer.

For example, if an ISP allocated 192.0.2.0/24 to a customer, the provider would have an ingress filter on the provider router side:

```
int s0
ip address 192.168.1.1 255.255.255.0
ip access-group 1 in

access-list 1 permit 192.0.2.0 0.0.0.255
```

As you can see, a standard access list is quite enough to implement ingress filtering.

IP Extended Access Lists

The standard IP access list is restricted to filtering based upon the source address of the packet. Extended IP access lists take the capabilities much further by allowing packets to be filtered based upon a variety of other factors, such as:

- Source IP address
- Destination IP address
- IP protocol
- Type of service field

The IP protocol field specifies the payload protocol being carried by the IP packet, such as TCP or ICMP. Note that application protocols such as FTP can be specified, but the IP payload type is TCP; there is an additional field that identifies the TCP or UDP payload.

IP Protocol Type

You can also use a protocol as a filter criterion. It can allow or disallow a variety of functions while still providing others. [Figure 1](#) shows a partial list of which protocols run over which.

See RFC 1700 for a list of protocol numbers and pointers to revisions.

ICMP Specific

For IP packets carrying ICMP, you can also specify the ICMP message type (e.g., Destination Unreachable) and, optionally, the message code (e.g., Communication Administratively Prohibited).

IGMP Specific

IGMP-specific information also can be specified, such as the message type (e.g., group report). Do be aware that controlling multicast access can be tricky. If you deny access to one host on a LAN by blocking its join requests, and another non-blocked host requests access to the same group, then the blocked host will hear any multicasts delivered to the LAN. Depending on the intelligence of the multicast router software, it may take more than one host requesting access to the group before the router sends multicasts rather than a unicast.

TCP and UDP Specific

You can specify decision criteria based on source and/or destination ports of TCP and UDP. An additional parameter, **established**, can be used with TCP and is discussed later.

See RFC 1700 for a list of port numbers and pointers to revisions. Ports are divided into three categories:

- Well-known (0-1023): IETF defined
- Registered (1024-2047): Not defined by IETF standards, but the IETF does document them when requested. Many Cisco proprietary protocols, such as SNA encapsulation, are in this range.
- Undefined (2048-65536): No standard value, usually for the client side of interaction

Do be aware that the "well-known" port associated with a particular protocol may be used only when setting up a connection. FTP, for example, has a PORT command that can redirect either the client or server to another address and port. See RFC 1579 for even more FTP port filtering issues.

FTP is not the only protocol that commonly redirects; HTTP does so routinely. To deal properly with such protocols, the router needs application protocol awareness, as in content-based access control.

For example, you may not want any Web access to the network. The Web is an allowable filter item; you have the option of filtering by port name or by port number. Let's look first at the filtering options:

Table 5. Filtering Options for TCP and UDP

| | |
|-------------|---|
| eq | Match only packets on a given port number |
| established | Match established connections |
| gt | Match only packets with a greater port number |
| lt | Match only packets with a lower port number |
| neq | Match only packets not on a given port number |
| range | Match only packets in the range of port numbers |

The interface can use these criteria to match against the port number of the packet. The two most common of these criteria are **eq**, which matches only packets on a given port number, and **established**, which matches only established connections.

Consider the following extended IP access list:

```
access-list 101 permit tcp any ...
 162.81.0.0 0.0.255.255 eq 80

access-list 101 permit tcp any ...
 162.81.1.2 0.0.0.0 eq 25

access-list 101 permit icmp any ...
 162.81.0.0 0.0.255.255
```

Leaking Routing Updates

It's often irrelevant whether you filter routing updates or not, because they usually have their time-to-live field set to 1, and will not go beyond the local router.

The first line configures access list 101 to permit TCP protocol packets (**tcp**) from any source address (**any**) to the destination on the network 162.81.0.0 255.255.0.0, provided the port number is 80. This criteria will allow Web access. The next line will permit TCP from any source only to the destination 162.81.1.2 and only to deliver SMTP packets; now, you can also have email delivered to your email server. The third line allows ICMP packets to be delivered from any source to any destination in the network. Finally, the implicit "deny all" statement will be there to stop any traffic that hasn't been permitted by one of these statements.

You now have available a number of different protocols to use as filter criteria. Let's assume you wish to allow only TCP from the nodes on segment 210.43.0.0. 255.255.0.0. The line would look like this:

```
access-list 104 permit tcp 210.43.0.0 0.0.255.255 any
```

This line permits (**permit**) any TCP protocol packet (**tcp**) from the source network 210.43.0.0 255.255.0.0 (**210.43.0.0 0.0.255.255**) to any destination address (**any**). Packets received from any other network or packets from this network communicating any other protocol would be discarded based on the implicit "deny all" at the end of the list.

Let's take a moment to look at a reverse of this setup. Let's assume that the only thing you wish to block is TCP protocol communications from this network of computers -- you want to allow all others to communicate. The lines would then read like this:

```
access-list 104 deny tcp 210.43.0.0 0.0.255.255 any
access-list 104 permit ip any any
```

Now, you're saying to deny any TCP protocol packets from the source network 210.43.0.0 255.255.0.0. If the packet doesn't match, then the next line item says to permit any IP protocol packet (**ip**) from any source (**any**) to any destination (**any**). The implicit deny all line would of course be next, but no packets would ever reach this point.

Advanced Mechanisms

Terms from computer science can be quite useful here. A stateless process is like traditional connectionless routing, or the decision process of ordinary access lists. Ordinary access lists make per-packet decision. In general, they retain no memory of any packets that have gone before.

Stateful mechanisms impose the additional overhead of memory, but also can make more intelligent decisions. A very basic stateful mechanism would block a transaction response unless the filter could match it to a query that previously was sent out.

One exception to the general rule that regular access lists are stateless is the **acknowledged** bit on TCP filters. This feature is not truly stateful, because it trusts the sender to have set the ACK bit properly. A true stateful access list, however, would know if a TCP connection really has been established.

Reflexive IP Access Lists

Reflexive IP access lists are used to configure IP session filtering. They are not a separate type of list, but are specified with special parameters in standard and extended IP access lists.

They provide the ability to filter packets based upon upper-layer protocol "session" information. This allows nodes to initiate sessions with nodes with which they would otherwise be unable to communicate and to maintain an open session through a restrictive router for the duration of the session. The main requirement of this type of "exception" to the access restrictions is that the session be initiated from inside the protected network. This command first appeared in Cisco IOS version 11.3 and is currently limited to use with IP filtered access lists only.

The command used to permit traffic through a reflexive access list is:

```
permit protocol any any reflect name
    [timeout seconds]
```

protocol -- This is the name or the number of the IP protocol that you will be allowing. You can use the keywords such as icmp, ip, nos, tcp, udp, or you can use the ip protocol number (0 to 255). The keyword **ip** is a match for any IP protocol.

name -- This allows you to specify a name for your reflexive access list. It can be up to 64 characters long but must begin with an alpha character.

seconds -- This is an optional command that allows you to specify how long the connection will remain open when session traffic is no longer being detected. If no timeout value is configured, then the reflexive access list will expire after the global timeout period.

A few main points to remember about reflexive access lists:

- This is only a permit command, there is no "deny" version.
- The session will only allow the same upper-layer IP protocol that was specified in the original packet transmission.
- The session will only be allowed between the source and destination addresses specified in the original packet.

Tip

Always be aware that a lack of communication can be caused by an access list in either direction. It might appear that you cannot reach a destination, because your **ping** times out, but the real problem is that the ICMP Echo Reply message is being blocked in the return path. With no reply, the **ping** will time out.

One way to test for this sort of problem is to run **debug ICMP** at the target router, and verify that you are or are not receiving ICMP Echo Request. Be extremely careful about enabling **debug** commands on any production router, because **debug** is very resource intensive. Don't forget that if you are telnetting into a remote router and want to see its **debug** output, you will need to use the **terminal monitor** command or have some other method of viewing logs.

- Traffic will cease 5 seconds after two set FIN bits are received, indicating the session is about to end. If no session traffic is detected for the time configured in the timeout period, the session will expire, and only the original source node will be able to re-initiate the session.
- Reflexive access lists do not contain the implicit "deny all" statement at the end.
- These lists are most often placed on border routers, those that pass traffic between internal and external networks.
- Reflexive access lists can be easy to mis-configure. The best approach is to keep them as simple as possible.

To create a reflexive access list,

1. Create and define the reflexive access list, let's call it "webaccess" and define the timeout period.

```
permit tcp any any reflect webaccess
ip reflective access-list timeout 90
```

2. Define the original access list:

```
access-list 106 permit ip 140.88.0.1 any
access-list 106 deny ip
140.88.0.0 0.0.255.255 any
access-list 106 permit ip
140.0.0.0 0.255.255.255 any
```

3. Insert the **evaluate** command into the access list

```
access-list 106 permit ip 140.88.0.1 any
access-list 106 deny ip
140.88.0.0 0.0.255.255 any
access-list 106 permit ip
140.0.0.0 0.255.255.255 any
evaluate webaccess
```

4. Apply the access list to the interface.

```
ip access-group 106 in
```

The command **evaluate** is used to tell the interface that if the packet has not received a match by the time it reaches that line in the access list it must now refer to the reflexive access list with that name.

Dynamic Access Lists

Closely associated with per-user authentication, dynamic access lists apply during the duration of a user connection session. As opposed to reflective access lists, they do not consider protocol interactions within a session, just the login and logout to an access server. They give the filtering capability of an extended IP access list.

Time Ranges in Extended Access Lists

In recent releases, you can also define access lists that apply during certain dates and times. This differs from dynamic access lists in that it applies to all traffic, not traffic from specific users.

Before using a time range in an access list, you need to define the range with an explicit **time-range** global command. You then refer to the name of this range from **access-list** rules.

Time ranges can be used with features beyond simple permit and deny operations. You can enable logging, for example, only at particular times of day. You could impose conditional quality of service.

Maps

One of the limitations of traditional access lists is that they allow only one operation once a pattern is matched. Maps have more capability, and are reminiscent of a programming language. The first application of maps were *route maps* that can permit/deny and also change fields in routing updates. Route maps are primarily used in BGP and policy routing but also can be useful in complex redistribution.

Other maps keep entering the IOS, such as crypto maps for IPsec.

Working with Access Lists

Now that you know how to configure and add access lists, we need to look at how you monitor the lists that we now have in place.

At times you will need to look at the access lists that have been configured on your device. Use the **show access-lists** command if you want to display all the access lists that the device has configured. The display for this command will look something like:

```

show access-lists
Standard IP access-list 3
  permit 46.8.191.3
  permit 46.8.191.4
Extended IP access-list 106
  permit eigrp host 46.8.191.18 any log
Novell access-list 918
  permit -1
  permit

```

When configuring an interface for a particular protocol, you may want to show only the access lists for that specific protocol. The command to show IPX access lists could produce an output such as the following:

```

show ipx access-lists
IPX extended access-list 900
  deny any 1
IPX sap access list Craft
  deny FFFFFFFF 107
  deny FFFFFFFF 301C
  permit FFFFFFFF 0

```

Naturally, the command **show ip access-lists** will provide only the access lists that are configured with the IP protocol:

```

show ip access-lists
Standard IP access-list 3
  permit 46.8.191.3
  permit 46.8.191.4
Extended IP access list 106
  permit eigrp host 46.8.191.18 any log

```

Should you desire to see the access lists that have been applied to a particular interface, use the **show {ip | ipx} interface s0** command:

```

show ip interface s0
Serial0 is down, line protocol is down
  Internet address is 126.1.1.100 255.255.0.0
  Broadcast address is 255.255.255.255
  Address determined by non-volatile memory
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is enabled
Outgoing access list is not set
Inbound access list is 1
  Proxy ARP is enabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachable are always sent
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is enabled
  IP multicast fast switching is enabled
  Router Discovery is disabled
  IP output packet accounting is disabled
  IP access violation accounting is disabled
  TCP/IP header compression is disabled
  Probe proxy name replies are disabled
  Gateway Discovery is disabled
  Policy routing is disabled

```

Finally, if you want a complete detail of the access lists that are configured on your device and the interfaces to which they have been applied, then use the **show running configuration** command (or "show run" is fine):

```

My Router#show running
Building configuration...

Current configuration:
!
version 11.0
service udp-small-servers
service tcp-small-servers
!
hostname IRVINE
!
enable secret 5 $1$hTLx$qB7V.cWAJM8aPTyBqQfrQ1
enable password two

```

```

!
!
interface Ethernet0
 ip address 10.0.0.1 255.0.0.0
 no mop enabled
!
interface Serial0
 ip address 126.1.1.100 255.255.0.0
 ip access-group 3 in
!
interface Serial1
 ip unnumbered Ethernet0
 shutdown
!
access-list 3 permit 46.8.191.3
access-list 3 permit 46.8.191.4
access-list 106 permit eigrp host 46.8.191.18 any log
access-list 918 permit -1
access-list 918 permit
snmp-server community public RO
!
line con 0
 password three
 login
line aux 0
 transport input all
line vty 0 4
 password three
 login
!
end

```

Tools for Troubleshooting Access Lists

Access lists can get quite complex and many things can go wrong when you try to configure and apply them. Let's go through a couple of the most common tools for tracking down access list problems.

Context-Sensitive Help is your Friend

The IOS can provide a lot of informational displays when you use the ? whenever you're unsure of the next item you might need or the options available to you:

```

access-list 1 permit ?
  Hostname or A.B.C.D  Address to match
  any                  Any source host
  host                 A single host address

```

```

access-list 801 permit ?
  -1                   Any IPX net
  <0-FFFFFFFF>        Source net
  N.H.H.H             Source net.host address
  <cr>

```

```

access-list 901 permit ?
  -1                   Any IPX protocol type
  <0-255>              Protocol type number (DECIMAL)
  <cr>

```

Note that the options displayed when you use the number 901 are different from those for 1 or 801.

Finally, always remember that there is an implicit "deny all" at the end of each list. If you only wish to restrict certain traffic through the use of "deny" statements, then you must include a **permit any** statement at the end of each list, or any traffic not specifically permitted in the list will automatically be discarded and your list won't work.

Logging with Access Lists

The IOS gives you the ability to track traffic that matches explicit permit or deny rules in either standard or extended IP access lists. To avoid overrunning the log, the first message that matches a rule causes a log message. When logging is in effect, after the first log event, the IOS captures logging information as 5-minute summaries for each source address. The information captured is:

- Number of the access list

- Permit or deny action
- Source IP address
- Count of packets permitted or denied in the 5-minute period.

When planning logging, do consider the possibility that an attacker will deliberately generate large numbers of violations in order to exhaust your log storage, and then launch the real attack. It is worthwhile to have a host script that monitors the rate at which the log is filling, and generate an alarm if the rate is unusually high.

Conclusion

Hammers are essential parts of any carpenter's tool chest. Proper tool chests, however, also contain saws, chisels, drills, screwdrivers, and, above all, measuring tools.

Access lists are an essential part of any Cisco tool chest. They are not the only tool you will need for security, performance tuning, etc., but any solution in those areas is very likely to include access lists or other, related pattern-matching functions such as priority lists and route maps.

Practically every group of access control lists has optional syntaxes and additional features that will continue to expand with every new IOS release.

References

RFC 2827 *Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing*. P. Ferguson, D. Senie. May 2000.

RFC 1579 *Firewall-Friendly FTP*. S. Bellovin. February 1994.

RFC 1812 *Requirements for IP Version 4 Routers*. F. Baker. June 1995.

RFC 2644 *Changing the Default for Directed Broadcasts in Routers*. D. Senie. August 1999.

NSA Aqua Book at <http://www.radium.ncsc.mil/tpep/library/rainbow/NCSC-TG-010.txt>

NSA Orange Book at <http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.txt>

NSA Red Book, <http://www.radium.ncsc.mil/tpep/library/rainbow/NCSC-TG-005.txt>

Access Lists Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Configuring an Access List for 5 different situations:

1. Configure an access list to allow all users in the enterprise to access the Time and Attendance application on TCP port 2200.
2. Configure an access list to allow administrators full access to the Las Vegas site.
3. Configure an access list to restrict all other access to the Las Vegas site.
4. Configure access control for the terminal lines on all routers.
5. Configure an access list to prevent ICMP "administratively prohibited" messages from being sent to hosts outside the corporate network.

Access Lists Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Access Lists Lab Scenario

[Introduction](#)
[Network Diagram](#)
[Lab Objectives](#)
[Solution](#)

Introduction

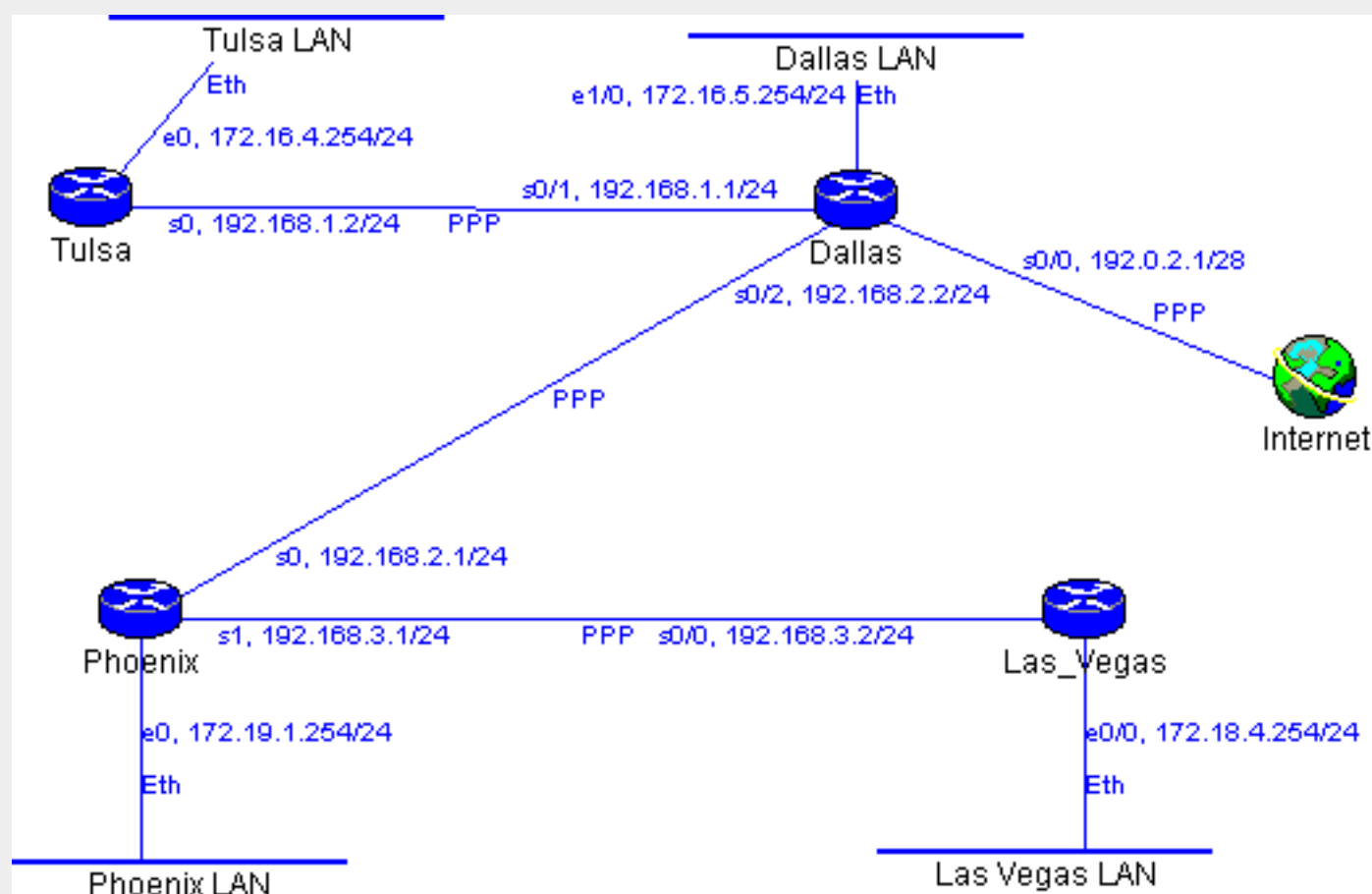
As network administrator for Galaxy One Inc., you are responsible for all routers and switches in the internetwork. The internetwork consists of four sites: Dallas, Tulsa, Las Vegas, and Phoenix. A drawing of the network is shown below. You must install all the network devices, configure them, and maintain them. It is also your responsibility to maintain connectivity across the corporate WAN and properly secure the network. Securing the network is one task that never seems to end.

Much of the work involved in securing the network stems from the ever-changing threat from entities outside your network, as well as the constantly changing political climate within your own organization. Now, management is at it again. They have decided that they are no longer satisfied with allowing full access to objects within the corporate network to all subjects within the internal organization. They have decided that certain objects should have controlled access, even for subjects that are known to be within the organization.

Specifically, they have decided that the resources in the accounting department, located at the Las Vegas site, should be off limits to all other organizations within the company, with the exception of the Time and Attendance application that every employee must access. Employees enter their timesheets electronically, and this information is transferred across the network to a database server in the accounting department. The client/server application that handles this operates over TCP using port 2200. Accounting staff members that are located outside Las Vegas, as well as system administrators, need full access to all of the resources in the Las Vegas site. These users all reside on the 172.16.4.0/24 network in Tulsa.

You decide to take this opportunity to control access to the router terminal lines, as well, in order to ensure that only designated administrators can gain remote access to the routers. You also want to implement a security measure that can prevent users from outside the organization from knowing that you have access control lists in place.

Network Diagram



Lab Objectives

1. Configure an access list to allow all users in the enterprise to access the Time and Attendance application on TCP port 2200.
2. Configure an access list to allow administrators full access to the Las Vegas site.
3. Configure an access list to restrict all other access to the Las Vegas site.
4. Configure access control for the terminal lines on all routers.
5. Configure an access list to prevent ICMP "administratively prohibited" messages from being sent to hosts outside the corporate network.

Solution

1. Configure an extended IP access list on the Las Vegas router. The list should contain the following entry to allow access to the Time and Attendance application:

```
access-list 101 permit tcp any 172.18.4.0 0.0.0.255 eq 2200
```

2. Add another entry to access list 101 on the Las Vegas router. The following entry will allow the administrators and accounting staff in Tulsa full access to the Las Vegas network:

```
access-list 101 permit ip 172.16.4.0 0.0.0.255 any
```

3. Without any additional entries, all other access to the Las Vegas site will be restricted by the implicit deny all at the end of access list 101. Apply this list as an incoming access control list on the Las Vegas router interface s0/0 using the following command:

```
Las_Vegas(config-int)#ip access-group 101 in
```

4. All administrators are located in Tulsa on network 172.16.4.0/24. Configure a Standard IP access list to allow access to the terminal lines only to that network:

```
access-list 10 permit 172.16.4.0 0.0.0.255
```

Apply this list to all terminal lines using the following commands on each router:

```
Tulsa(config)line vty 0 4  
Tulsa(config-line)access-class 10 in
```

5. Create an Extended IP access list on the Dallas router that prevents ICMP "administratively prohibited" messages from being sent out over the connection to the Internet:

```
access-list 102 deny icmp any any 3 9  
access-list 102 deny icmp any any 3 10  
access-list 102 permit ip any any
```

Apply access list 102 as an outbound access control list to the Dallas router interface s0/0 with the following command:

```
Dallas(config-int)#ip access-group 102 out
```

Access Lists Labs

Basic Router Operation Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Basic Router Operation (IOS)

The purpose of this tutorial is to introduce you to Cisco router operation. This tutorial will also help you study for several topic areas that may be tested in Cisco certification exams including IOS basics, Router CLI, and troubleshooting.

This tutorial is divided into four sections: Overview of Cisco Router Hardware and Software, Basic Router IOS, Basic Router Configuration, and Basic Router Maintenance and Troubleshooting. There are also Appendices that include review questions (with answers) and additional material that you may find useful.

350-018 Labs

Basic Router Operation

Tutorial
Lab Abstract
Lab Scenario

Basic Router Operation Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Basic Router Operation

[Introduction](#)

[Overview of Cisco Router Hardware and Software](#)

[Hardware and Software Components of the Cisco 2501 Router](#)

[RAM, ROM, NVRAM, and Flash in Cisco Routers](#)

[Router Boot Sequence](#)

[IOS Options](#)

[Basic Router IOS](#)

[Router Access through the Console Port](#)

[Router Access through Telnet](#)

[Command Interpreter](#)

[User Mode](#)

[Privileged Mode](#)

[CLI Help](#)

[Inline Help -- Words](#)

[Inline Help -- Command Syntax](#)

[Command Line Completion](#)

[Syntax Checking](#)

[Hot Keys Used for Editing](#)

[Advanced Editing](#)

[Command History](#)

[Basic Router Configuration](#)

[Setup Mode](#)

[Manual Router Configuration](#)

[Setting Router Passwords](#)

[The Console Password](#)

[The Enable Password and Enable Secret Password](#)

[The VTY Password](#)

[The Auxiliary Line Password](#)

[Configuring an IP Address](#)

[Configuring Banners](#)

[Committing Configuration Changes to NVRAM](#)

[Configuring Clock Rate](#)

[Basic Router Maintenance and Troubleshooting](#)

[Backing up Router Configuration Files](#)

[Backing Up IOS Software Images](#)

[Show Commands](#)

[Basic Show Commands](#)

[Show Version](#)

[Show Interfaces](#)

[Show Protocols](#)

[Show Flash](#)

[Advanced Show Commands](#)

[Show Memory](#)

[Show Processes](#)

[Show Stack](#)

[Show Buffers](#)

[Show Processes CPU](#)

[Show CDP Neighbors](#)

[Debug Commands](#)

[Fallback](#)

[The Configuration Register](#)

[Changing the Configuration Register](#)

[Booting from ROM \(Boot Field = 01\)](#)

[Booting from Flash \(Boot Field = 02\)](#)

[Appendix A. Review Questions and Answers](#)

[Appendix B. Cisco Router Series](#)

[Series 700](#)
[Series 1600](#)
[Series 2500](#)
[Single LAN](#)
[Dual LAN](#)
[Router/Hub Combo](#)
[Access Servers](#)
[Series 2600](#)
[Series 3600](#)
[Series 4000](#)
[AGS+ and Series 7000](#)
[Series 7100 and 7200](#)
[Series 7500 and 10000](#)
[Series 12000](#)

Introduction

Today Cisco Systems(r) has become the world's foremost developer and manufacturer of internetworking equipment and software. Cisco(r) develops and manufactures over 80% of the routing equipment that controls the flows of information traveling on the Internet and private internetworks.

With this market dominance comes a huge demand for engineers and administrators that understand Cisco's routers and other products. One way that you can demonstrate understanding of Cisco routers and internetworking fundamentals is to pass the Cisco Certified Network Associate (CCNA(tm)) exam (is not associated with Cisco). In July 2000 Cisco retired the CCNA 1.0 exam (640-407) and in August began offering the CCNA 2.0 exam (640-507). The new exam has a new list of objectives that are broader in scope than previous objectives and in fact are no longer really objectives but recommended topic areas for study. You can find these new topic areas here:

http://www.cisco.com/warp/public/10/wwtraining/certprog/testing/pdf/ccna_507.pdf. (is not associated with Cisco.)

The purpose of this Tutorial is to introduce you to Cisco router operation. This paper will also help you study for several topic areas that may be tested in the CCNA 2.0 exam including IOS basics, Router CLI, and troubleshooting.

This paper is divided into four sections: Overview of Cisco Router Hardware and Software, Basic Router IOS, Basic Router Configuration, and Basic Router Maintenance and Troubleshooting. There are also Appendices that include review questions (with answers) and additional material that you may find useful.

Overview of Cisco Router Hardware and Software

Hardware and Software Components of the Cisco 2501 Router

You could say that a router is nothing more than a small PC with a smaller operating system and no direct user interface hardware, such as keyboards or video monitors. If you look at what a router does for networking, it is essentially the same as a personal computer. A router looks and acts like a PC in many ways. Like a PC, the router is built with input/output (I/O) ports, it has a processor and memory chips, it provides a set of instructions that tells the router what to do, and it has an operating system that runs the router.

This paper will focus on the components and functions of one of Cisco's entry-level router models, the Cisco 2501. Throughout this text when a router configuration or setup is described we will be speaking of a 2501 unless the statement specifically refers to another router model or number. The 2501 router is a member of the 2500 series family of Cisco routers. It has a single Ethernet interface and two serial interfaces and is powered by a Motorola processor running at 25MHz. The 2501 router is the router of choice for most people getting started with hands-on Cisco router experience; however the Cisco 1600 router, with only one serial port, is adequate and costs a lot less. You can purchase refurbished 2501s for around \$900 from various Web auction sites or network hardware resellers that specialize in Cisco equipment. The Cisco 1600 router should cost about \$700.

Let's take a closer look at the Cisco 2501 starting at the back of the router (see Figure 1). On the far left is one 10 megabit Ethernet AUI connector used for LAN connections. You will need an Ethernet transceiver/adaptor attached to this port so that you can change this port from AUI to RJ45 and make it Category 5 compliant for a typical 10baseT or 100baseT network cable. To the right of the AUI connector are two high-density 60-pin serial connectors. These serial connectors are used for WAN connections. Next to the serial ports are interfaces marked CONSOLE and AUX. Both of these interfaces look like telephone wall jacks that you plug your phone into. We'll discuss the purpose of these ports soon. To the right of the AUX port is small green LED. If this light does not come on when the router is turned on, your router may have a bad memory card or processor. On the far right is the connector where the power cord plugs in.

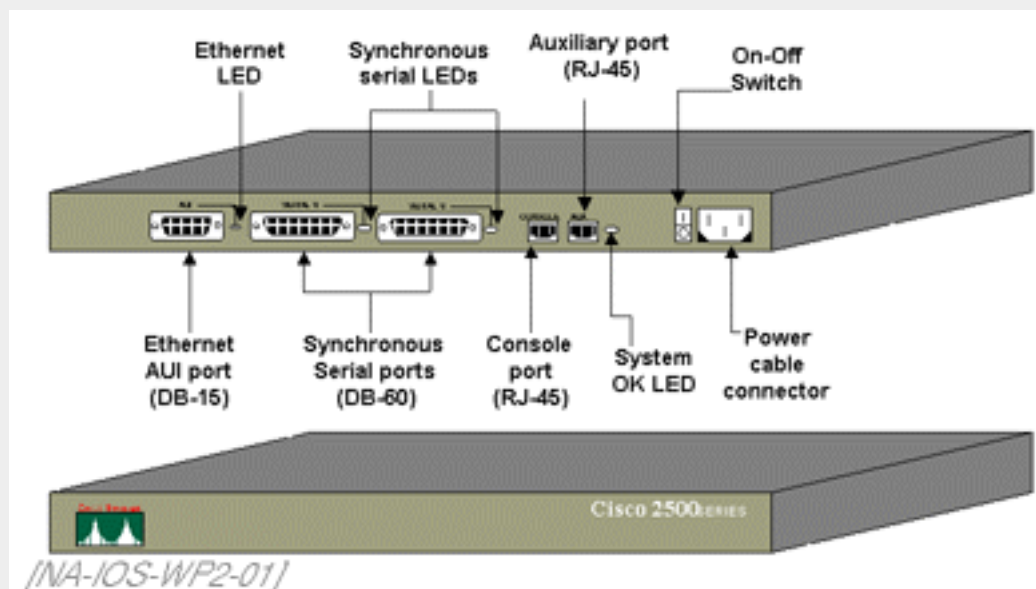


Figure 1. Cisco 2501 Router Front and Rear Views

As you can see, the 2501 router has three types of interfaces. Interfaces are where the router meets the outside world and are the means by which a router receives or sends information. Serial interfaces are mostly used to connect long-distance as in a WAN (Wide-Area Network). Later in this paper we will describe how to connect routers together using their serial ports via DTE/DCE cables to simulate various WAN connections. Besides Ethernet, you can have other LAN interfaces on a router, such as Token Ring or FDDI (Fiber Distributed Data Interface) interfaces. For example, the Cisco router 2502 has two serial interfaces and one Token Ring interface instead of an Ethernet interface. See Appendix B for descriptions of various Cisco routers and the type of interfaces they have.

There are two kinds of interfaces on a router, fixed interfaces and modular interfaces. The three interfaces we just described are fixed interfaces -- they are connected directly to the motherboard and can't be removed. Modular interfaces can be dynamically added or removed by plugging add-in modules or cards into the modular router bus interface. In Cisco routers, the type of bus depends on the model (or family) of router. In a 3600 router, the bus is a PCI bus; in a 7200, it is dual independent PCI buses; in the 7500 series, it is a CyBus. For more information about these types of bus architectures, you can consult the [Cisco web site](#) (is not associated with Cisco.)

A name and a number denote each of the interfaces on a Cisco router. The first of any of the interface types starts numbering at zero instead of one. For example, the 2501 router has two serial interfaces and one Ethernet, so we would have **interface serial 0**, **interface serial 1**, and **interface ethernet 0**.

RAM, ROM, NVRAM, and Flash in Cisco Routers

In order to understand what a router does, it helps to know where the basic components are found on the motherboard. In the following paragraphs we will describe the physical characteristics and the position of each component and give a brief explanation of each component's function.

There are three main parts of a router: central processing unit (CPU), memory, and interfaces. The CPU is basically the same as that of a PC; it controls the execution of commands and instructions and directs the flow of information inside and out of the router. The memory comprises four different memory elements: Random Access Memory (RAM), Read-Only Memory (ROM), Non-Volatile RAM (NVRAM), and Flash memory.

Notice the position of the several memory cards and chips in Figure 2. On the left, you can have up to two Flash memory cards. The Flash memory is where the IOS images are located. Cisco routers are almost completely useless without the Cisco IOS, also known as the system software image. Flash memory is a type of erasable, programmable, read-only memory and is available on most Cisco routers as a Flash memory chip, and on some models as a PCMCIA Flash card.

The interfaces discussed show how on a Cisco 2500 series router each interface is designated by interface type and number, such as ethernet0. On newer, modular routers and switches you will run into other interface notations. If the device is modular, it will have slots, cards, and ports on those cards, and you will run into interface notations such as ethernet 1/0 or serial 2/1/1. Remember that all slots and interfaces start at 0 and count up from there. In the case of ethernet 1/0, this represents the first port on the second slot interface card. This two-level representation is seen on all 2600 and 3600 routers, as well as on Catalyst 5000 series switches.

Cisco 7200, 7500 and 12000 routers can have cards called Versatile Interface Processor that can accommodate multiple port adapters on a single slot. Where you have multiple port adapters with multiple ports in a single slot, you will see a three-level notation. Where you see a three-level notation, such as serial 2/1/0, this represents the first port on the second card in the third slot. Each level simply means a more granular description of which port you are looking at. As you work on more routers, you will encounter this more frequently, and it will become more apparent. As the 2500 series routers become more frequently replaced by the 2600 series routers, this will become commonplace notation.

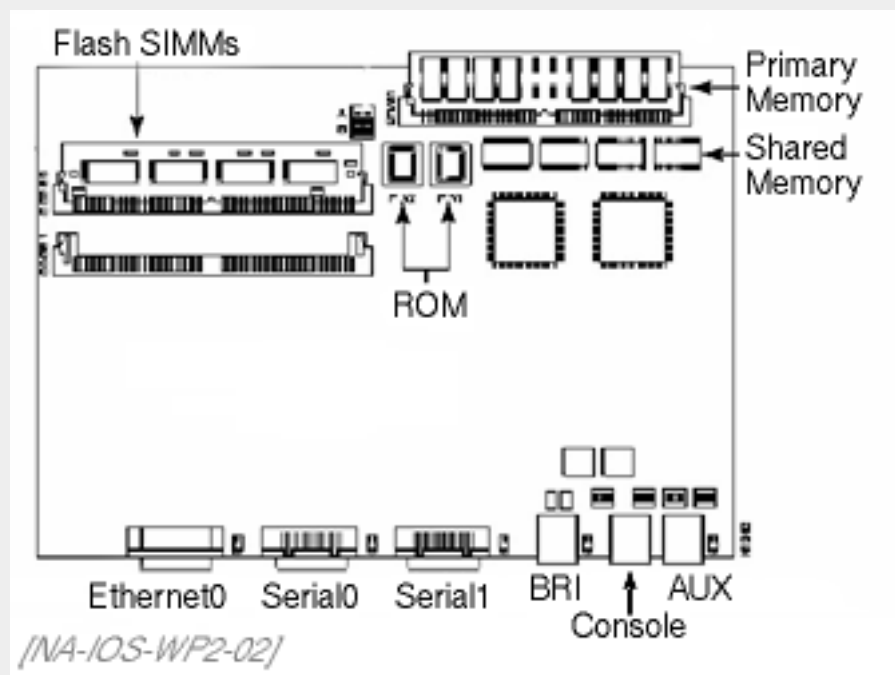


Figure 2. Cisco 2503 Motherboard

Situated at the top-right is the Primary memory, which is a DRAM SIMM memory card, and right below this card are soldered chips called shared memory. There are two types of DRAM memory in the Cisco 2500 series routers: primary and shared (packet). Primary memory is used to store the operating configuration, routing tables, caches, and queues. Shared memory is used to store incoming and outgoing packets. If you have an extra 16 or 8 MB SIMM card lying around from an old 486, it should work fine as an upgrade for Primary memory.

The RAM we just talked about is used strictly for running operations and will get erased with every power off or reload. Router RAM is used just like the RAM in your PC. It stores the running configuration and is where all working information is stored. The running configuration is a partner to the startup configuration. After a router boots, the startup configuration is delivered to the running configuration and it's with this configuration that you typically work to make changes. After you're finished with the running configuration, you then save your changes to the more permanent startup configuration. If you could view the RAM after the router starts up, what you would see is the following: a cached subset of operating system commands, a copy of the startup configuration for running access, all route tables and anything else that dictates how the router behaves.

NVRAM is different from RAM. The contents of NVRAM can be changed, modified, or erased at the user's command. NVRAM will not lose its contents when the router is turned off. The startup configuration file is stored in NVRAM or on the network. The startup configuration file is the instruction set the router uses to boot with. NVRAM is not in a socket. It is part of the motherboard, which makes it difficult to upgrade and hard to find.

Both Flash and Primary memory can be upgraded by simply snapping in new cards. The amounts of Flash memory and Primary memory are typical configuration elements that describe a particular 250x router when advertised. You may see or hear of a router advertised as having an 8 by 8 configuration. This means that the router has 8 MB of Flash RAM and 8 MB of Primary memory.

The Cisco router ROM is stored on memory chips that are located on the motherboard. Just underneath Primary memory are two ROM chips. Much like in a PC, the ROM stores the most elemental functions a router must perform to begin operation. ROM is a form of permanent memory used by the Router to store the "Power-On Self-Test" that checks the Router on boot up and the "Bootstrap Startup Program" that gets the Router going. In addition, ROM contains a very basic form of the Cisco IOS software, which is used during certain occasions. In order to upgrade ROM you have to remove and replace chips. The two ROM chips easily pop right out.

Router Boot Sequence

It is important to know the router's boot process. In the event of a boot problem, you may need to spot where the problem is occurring in order to correct it. When a router is powered up, there is a predefined sequence of events that must take place for the router to complete the booting process. First is the Power-On Self-Test (POST), where a test routine is run on the CPU, memory, and interface electronics to make sure there are no circuitry problems. The boot sequence steps are listed below:

1. The "Power-On Self-Test" checks the Router Hardware. This includes the CPU (Central Processing Unit), memory, and interfaces.
2. The "Bootstrap Program," which is stored in ROM, runs itself.
3. The "Bootfield" from the configuration register (discussed later) is read to find out the proper Operating System source.
4. The "IOS software image" is loaded into RAM. The IOS software image can be loaded from Flash, TFTP, or ROM.
5. The Startup Configuration File is read from NVRAM or a TFTP server and then loaded into the RAM. The Configuration File is then executed one line at a time and starts the processes to run the router according to that file.
6. If no "Startup Configuration File" is found in NVRAM, the Cisco IOS will offer you the chance to use the "System Configuration Dialog" or commonly called the "Setup Script." This is a set of questions for you to answer to create a basic configuration.

To see how the boot sequence relates to the four types of memory we discussed earlier, refer to Figure 3. Some of the descriptions in the figure may contain terms you're not familiar with yet, but we'll explain those later in the paper. It is important to understand the relationship between memory and boot sequence for troubleshooting purposes, which we'll also discuss later.

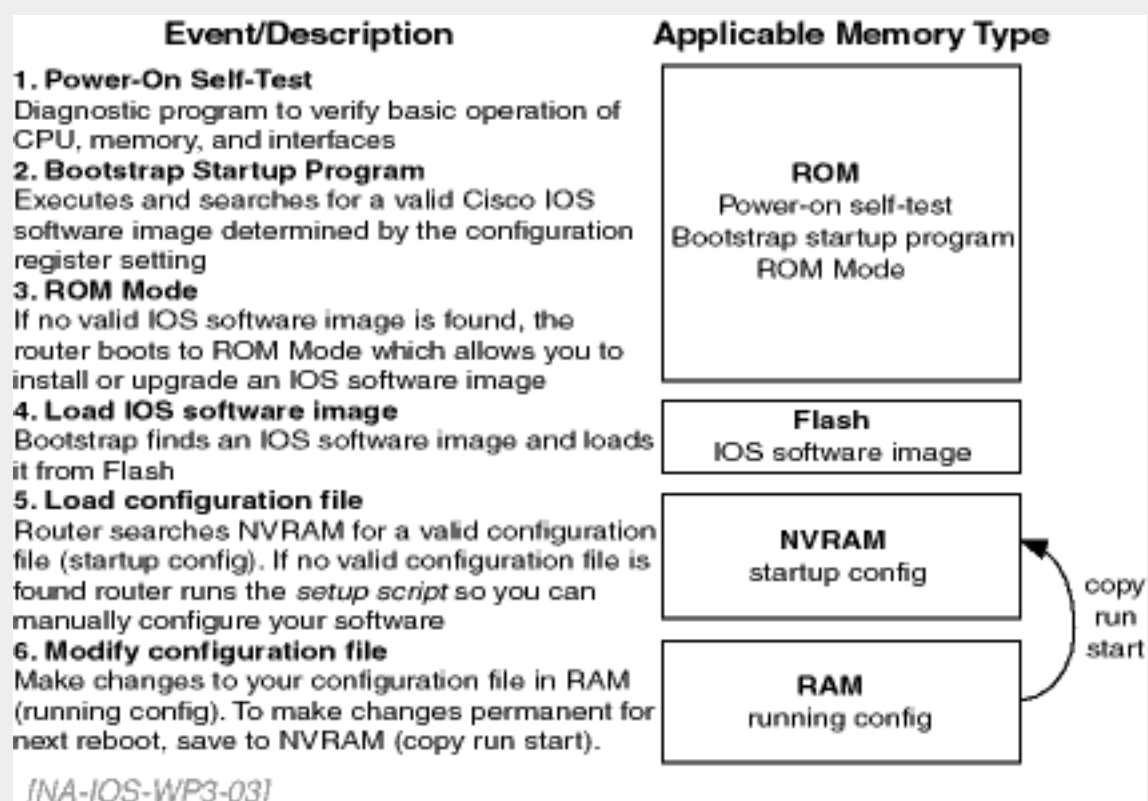


Figure 3. Router Memory Types and Their Functions

IOS Options

Cisco is a hardware vendor company, but if you ask anyone at Cisco, they will likely tell you they are a software company. Their hardware is nothing more than an expensive boat anchor without the IOS software. Knowing this, we can see why knowledge of IOS system software manipulation is crucial to success in working with Cisco routers.

Cisco routers have evolved much over the past 15 years. The IOS software has evolved along with them. In this evolution, Cisco has incorporated different features and functionality with every new release and version. IOS software images are bundled based on feature sets. Each feature set contains support for a certain protocol, group of protocols, or added feature. Some examples include the Desktop feature set that bundles most of the basic LAN networking protocols together, such as IP, IPX, AppleTalk, DECnet, bridging, WAN protocols, etc. Other feature set capabilities you may wish to implement include:

- IP means the router can manage protocols for the Internet.
- IPX means the Novell protocols can be handled.
- AT stands for the AppleTalk protocol for Macintoshes.
- DEC stands for the Digital Equipment Corp. protocols.
- APPN is for the Advanced Peer to Peer Networks (IBM).
- PLUS means NAT (Network Address Translation) can be performed.
- IPSEC is an Internet SECURITY feature (encryption) usually not found or needed in typical WANs.
- RAS is an alternate security solution.
- FW means there are firewall capabilities built into the IOS.

Basic Router IOS

Let's now cover how we access the Cisco router and its operating system through the user interface (UI).

Router Access through the Console Port

Since a router doesn't come with a video monitor, you need to use your computer monitor as the router's screen. When you use your monitor like this you call it a terminal. The terminal has an interface called the user interface (UI), which is text command line based instead of a mouse-operated graphic user interface (GUI).

First, let's look at how we can access the UI on a Cisco router. If you receive a Cisco router in the box, there will be a couple of things that accompany the router itself. There will be a black cable called a console cable. If you don't have the black cable, you can use any category 5 cable. Both ends of the black console cable have an RJ45 pin, which looks like a phone plug. Also included is a 9-pin or 25-pin serial adapter to be attached to one end of the black cable that you then connect to the serial port on your computer. The RJ45 end of the cable goes into the Console port on your router.

Now you're ready to deal with the terminal program that will allow your computer to become the router's terminal window. You will need to use some sort of terminal emulation program, such as Hyper Terminal for Windows 95. If you have not used Hyper Terminal, click Start->Programs->Accessories->Hyper Terminal. Once the folder with Hyper Terminal comes up, double click on Hypertrm.exe. As shown in Figure 4, Hyper Terminal will come up and ask you to enter a name for your connection. Type in **direct to com1** and hit Enter.



Figure 4. Name Your Connection

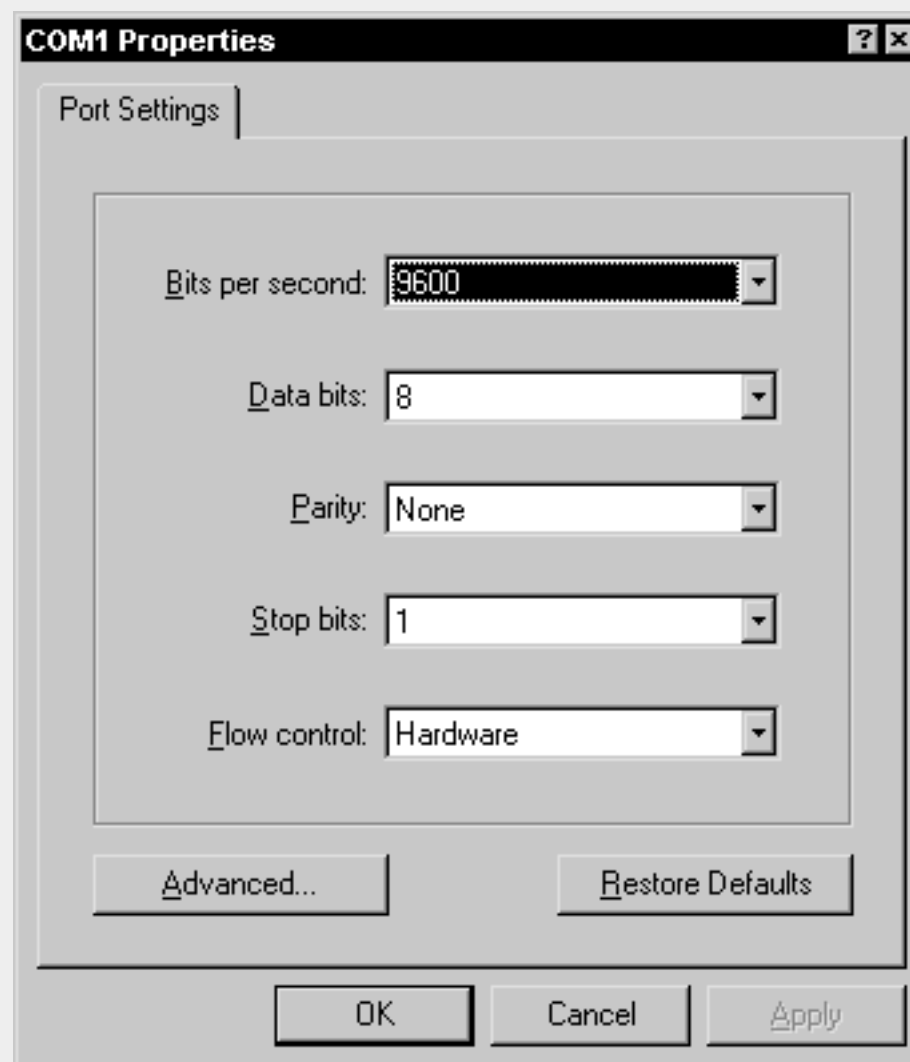
On the next screen, there will be a drop-down list box titled "Connect using." Click on the arrow at the right and choose **Direct to com1** and hit Enter.



[NA-IOS-WP2-04]

Figure 5. Enter Connection Method

The next screen will be the settings for how the terminal emulation program communicates with the console port on the router. The settings should be as shown in the following diagram, bits per second is set to 9600, data bits is 8, parity is none, stop bits is 1, and flow control is hardware. Click OK.



[NA-IOS-WP2-05]

Figure 6. Enter Settings for the Serial Connection

Once you are done with your session, you can close Hyper Terminal. When you do, you will be asked if you want to disconnect. Answer "yes." After this, you will be asked if you would like to save the session. You can go ahead and save this if you'd like, or you can say "no." If you say "no," however, you will have to go through this configuration again every time you go into Hyper Terminal. If you do save the session, when you bring up Hyper Terminal from the start menu, there will be an icon in that folder that will act as a shortcut to get around having to do the configuration steps every time.

Router Access through Telnet

Once a router has been initially configured and has basic network connectivity, the UI can be accessed remotely via Telnet instead of just through the console port. To Telnet from a Windows 95/98 or NT machine, click Start->Run. When the Run box comes up, type in the word "Telnet," and hit Enter. This will run the basic Telnet utility that comes with Microsoft Windows products. Once the Telnet utility program has come up, click **Connect** from the menu, then **Remote System**. Enter the IP address of the remote router to which you wish to Telnet and hit Enter. From there on, the interface will be mostly the same as a console connection. We will cover this again after we show you how to perform initial configuration on a Cisco router. Telnet access to any router can only be accomplished after initial configuration has been performed (from the Console port).

Command Interpreter

Once the router is booted, the Cisco IOS command interpreter, called the Exec, is ready to accept your commands. If you are familiar with DOS, the Exec is much like COMMAND.COM. In the Exec command interpreter, there are two modes of operation: user mode and privileged mode.

User Mode

User mode is denoted by a greater than (>) sign after the router prompt, like this:

```
Router>
```

It allows the user to enter and execute some limited and basic monitoring commands. For example, you can only do things like view basic router information, check status of the router and routing tables, and test simple connectivity. There are no configuration permissions and only limited troubleshooting commands available in user mode.

Privileged Mode

Privileged mode is denoted by a pound (#) sign after the router prompt, like this:

```
Router#
```

In privileged mode, the user has access to all commands available in user mode, all commands necessary to troubleshoot and debug, as well as access to router configuration mode, which is where all router configuration commands are entered. We discuss router configuration mode later in this chapter.

Commands in the Exec are entered via the Command Line Interface (CLI). This is not a new or different access level with its own prompt, rather it's the set of tools associated with the Exec modes.

CLI Help

The CLI has some functions that will give the user a little extra help in entering commands, troubleshooting, and configuring the router.

An editing feature of the CLI is that you can arrow back and forward on a line to edit misspellings. There is one caveat, however. The backspace and delete keys do the same thing in the Cisco CLI.

If you have access to a Cisco router and tried these commands, you probably noticed the **--more--** notation when you entered the **show version** command.

You have three options when you see **--more--**:

- If you press the [space] bar, the command interpreter will display another full screen of information.
- If you press the Enter key, you will get one more line of output.
- If you want to exit before seeing the rest, you can press any other key. The [q] key is most often used.

Inline Help -- Words

Another CLI feature that is a very helpful tool is inline help, also known as the question mark, which provides us with context-sensitive help. Context-sensitive help can be used in two ways, command syntax and word help. Let's try one of the above commands on a Cisco router and see what the ? will do for us:

```
Router# show v?  
version vines vpdn  
Router# show v
```

This is called word help. The output will be a line of data with several possible commands to complete the word that starts with the letter v. As you can see, the outcome was **version vines vpdn**, but what you might not notice is the second **show v** on the next command line. When using context sensitive help, the CLI will automatically repeat the command to where you left off to save you from having to re-enter that text. When using word help, make sure to fill in as many of the letters as you can, then immediately follow that with the question mark (?). Make sure not to leave a space. This inline help can be used in all levels and positions of a command.

Inline Help -- Command Syntax

Help with command syntax can be attained from the question mark as well. If you are configuring the IP address of an Ethernet interface, but are not sure of the syntax, you can use the (?) to help you along:

```
Router(config-if)# ip add ?  
A.B.C.D IP address  
Router(config-if)# ip add 192.168.1.1 ?  
A.B.C.D IP subnet mask  
Router(config-if)# ip add 192.168.1.1 255.255.255.0 ?  
secondary Make this IP address a secondary address
```

<cr>

Here we used the command syntax help to get us all the way through setting the IP address for the ethernet0 interface. You will notice that each step along the way gets us a little closer to where we want to be, and each step gives us a little different information. The last step shown with a question mark shows a couple of responses. The key one to notice here is that the <cr> symbol means that we have fulfilled the necessary command requirements and we can now simply hit the Enter key to execute it. (The <cr> stands for carriage return, i.e., the Enter key).

Command Line Completion

Another CLI feature is command line completion, the function of the [tab] key. Let's take the **show version** command and try it again, but this time let's put the [tab] key in the place of the question mark:

```
Router# show v[tab]
Router# show ve[tab]
Router# show version
```

The tab key will fill in with the command that matches the text you enter. If the amount you typed isn't enough, like the first line above, the CLI will make a sound and do nothing but duplicate what you have typed on the next line. This means that the command was too ambiguous, in other words there is more than one command that starts with the typed letter(s).

The Cisco router IOS is actually derived from a Unix operating system kernel. From this origin, the Cisco CLI gained the ability to accept truncated commands. As long as the truncated command you enter is enough of the command to distinguish it from any other command with similar text, the CLI will accept it and the Exec will process it. As an example, let's look at the **show version** command from earlier. To accomplish the same thing, you could also type in **sh ve**. This is an easy way to get what you want done while conserving the maximum number of keystrokes, and as every Unix-head knows, that is the key to slowing the rapid expansion of the universe. In the text of this Tutorial, you will see terse versions of each command, but if you would like to see the full command, get some time in front of a router and hit the [tab] key a lot.

If you truncate the command you are typing too much, and there is a command that has the same beginning, you will see the following error:

```
Router# con
% Ambiguous command: "con"
Router# con?
configure connect
Router# con
```

As you can see, there are two commands that begin with "con," and you must specify which one you wish to use. In this case you can type "con?" and get the two options available to you.

Syntax Checking

Automatic syntax checking is built into the CLI. If a command is improperly spelled, or is not a valid command, the router will respond by placing a caret symbol below the errant letter, word, or argument. If you were to type in **show versoin** like this example, here is what you would receive in response:

```
Router# show versoin
                ^
% Invalid input detected at '^' marker.
```

Hot Keys Used for Editing

Hot keys are built into the CLI editor to help with simple editing functionality. If you are familiar with Unix, you will recognize quite a few of these:

Table 1. IOS CLI Hot Keys

| Hot Key | Function |
|-----------|---|
| Delete | Removes one character from the right of the cursor. |
| Backspace | Removes one character from the left of the cursor. |
| Tab | Fills in remaining text of a partial command. |
| Ctrl-a | Moves to beginning of current line. |
| Ctrl-r | Redisplays a line. |
| Ctrl-u | Erases all characters on current line from cursor left. |
| Ctrl-w | Erases word (all characters left of cursor up to next space character). |
| Ctrl-z | Ends config mode (same as end command in config mode). |
| Up arrow | Scrolls through previously entered commands. |

Advanced Editing

If the end of a line goes too long, it will not automatically wrap to the next one. Instead the Cisco IOS command shell gives you a dollar sign (\$) at the beginning or end of the line. This indicates that you are an over-achiever and have typed too much, at least too much to be shown on the screen. If you type in a very long command, your line would now look like this:

```
Router#$ this is a way too long line that is full
```

Note that the \$ goes after the Router Prompt. If you keep typing the line will shift over as you type, hiding more of the beginning of the sentence.

```
Router#$ is full of sound and fury, signifying nothing!
```

You can get back to the beginning of your novel by pressing [CTRL-A], and this would be the effect:

```
Router# For Demo Only this is a long line that is full $
```

If you want to you can turn off these Advance Editing Tools by simply typing in **terminal no editing** at the prompt. A reason to turn off the Advanced Editing is that the tools are often incompatible with computer-executed scripts. Since this would be a silly thing to do if you are typing things in yourself, please turn them back on by typing in **terminal editing**.

Command History

The CLI keeps a history of the most recent commands entered, accessible by pressing the up arrow. For an example, let's say that a user entered the following three commands:

```
show version
show clock
show user
```

If that user decided that she wanted to show the version once again, she could press the up arrow key three times, and that command would show up on the CLI. Now just press the Enter key.

The Router keeps the last 10 commands you issued in its HISTORY, which is a special memory buffer that holds the "Command History." If you are using the VT-100 Emulator we talked about before, simply do the following.

- Press the UP Arrow key to go to the next most recent command.
- Press the DOWN Arrow key to go back down through the previous commands (after pressing UP arrow).

If you are a poor unfortunate without VT-100, you can use these instead:

- CTRL-P takes you to the "Previous" command.
- CTRL-N takes you to the "Next" commands.

Typing the command **show history** at the prompt gives you the list of the last 10 commands you have entered.

```
Router# show history

1. Command One
2. Command Two
3. Command Three
4. Command Four
5. Command Five
6. Command Sixx - (with a mistake!)
7. Command Six - (fixed now)
8. Command Eight - "There is No Command 7!"
9. Command Nine
10. Command Ten
```

You can increase the size of your History buffer by using the command **terminal history size**. The command below would give you 99 commands to play with.

```
Router# terminal history size 99
```

Basic Router Configuration

There are two ways to configure a router, manually and through the Setup script. If you have a router that has never been turned on or has recently

had the configuration file erased, the router will launch the Setup script, also known as the System Configuration Dialog (much easier to say "Setup script"). Manual configuration is quicker and more flexible, but the Setup script is easier for beginners because it steps you through the whole configuration process. Let's walk through a Setup script as if we have just booted a new router that has never been configured before.

Setup Mode

Setup mode is intended only for minimal configuration of an out-of-the-box newly arrived router. Do not fall into the trap of using it routinely. Almost any real-world configuration will require configuration features that are not available in setup.

The first part of the data capture here is from the system boot process:

```
System Bootstrap, Version 11.0(10c), SOFTWARE
Copyright (c) 1986-1996 by cisco Systems
2500 processor with 14336 Kbytes of main memory
```

Notice: NVRAM invalid, possibly due to write erase.

```
F3: 8022188+98780+316356 at 0x3000060
```

Restricted Rights Legend

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) of the Commercial Computer Software - Restricted Rights clause at FAR sec. 52.227-19 and subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS sec. 252.227-7013.

```
cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134-1706
```

```
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
RELEASE SOFTWARE (fcl)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
Image text-base: 0x03040270, data-base: 0x00001000
cisco 2500 (68030) processor (revision N) with
14336K/2048K bytes of memory.
Processor board ID 06160684, with hardware revision 00000000
Bridging software.
SuperLAT software copyright 1990 by Meridian Technology Corp).
X.25 software, Version 2.0, NET2, BFE and GOSIP compliant.
TN3270 Emulation software.
1 Ethernet/IEEE 802.3 interface(s)
2 Serial network interface(s)
32K bytes of non-volatile configuration memory.
16384K bytes of processor board System Flash (Read ONLY)
```

Note that we have been given a notice that NVRAM is invalid. This is not a problem since this router has never been configured. All it means is that NVRAM is empty, and there is no configuration to run from. That being the case, the Cisco router defaults to running the Setup script. As the system configuration dialog prompts you, there will always be a default answer to the question being asked in [brackets]. If this is the answer you want, you can just hit the Enter key.

Let's continue:

```
Notice: NVRAM invalid, possibly due to write erase.
--- System Configuration Dialog ---
```

```
At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.
Would you like to enter the initial
configuration dialog? [yes]: y
```

```
First, would you like to see the current
interface summary? [yes]: y
```

```
Any interface listed with OK? value "NO" does
not have a valid configuration
```

```
Interface  IP-Address  OK?  Method  Status  Protocol
Ethernet0  unassigned NO   unset   up       up
Serial0    unassigned NO   unset   down     down
Serial1    unassigned NO   unset   down     down
```

Configuring global parameters:

```
Enter host name [Router]: Router
```

The enable secret is a one-way cryptographic secret used instead of the enable password when it exists.

```
Enter enable secret: cisco
```

The enable password is used when there is no enable secret and when using older software and some boot images.

```
Enter enable password: cisco2
Enter virtual terminal password: cisco
Configure SNMP Network Management? [yes]: n
Configure LAT? [no]: n
Configure AppleTalk? [no]: n
Configure DECnet? [no]: n
Configure IP? [yes]: y
Configure IGRP routing? [yes]: n
Configure RIP routing? [no]: y
Configure CLNS? [no]: n
Configure IPX? [no]: n
Configure Vines? [no]: n
Configure XNS? [no]: n
Configure Apollo? [no]: n
Configure bridging? [no]: n
```

Let's summarize what the Setup script has asked and told us so far.

The first thing it asked is if we want to enter the Setup script.

Then the script asked us if we'd like to see an interface summary. This is not a necessary evil on a 2500, but could be useful with a 3600 or higher router that has a flexible configuration.

Now the Setup script asked us for the router name. This will be the name of the router that you see at the IOS prompt. Do not confuse this with a DNS name -- this router name will be locally significant only.

Next it asked us for the enable secret and the enable password. The reason the Setup script does this is stated in its block of text above, that some older software images cannot understand the encryption used on enable-secret passwords. This problem is prevalent in cases where people or businesses are using older routers that have older IOS images in the boot ROMs of their routers.

Next we will be prompted for the virtual terminal password, which is long hand for "Telnet password."

Configure SNMP, LAT, AppleTalk, DECnet, CLNS, IPX, XNS, Apollo, bridging are all other routable protocols that can be configured through the Setup script. For this example, we are only going to set up IP with RIP routing protocol.

Now we will move on to configuring the physical interfaces on the router:

Configuring interface parameters:

```
Configuring interface Ethernet0:
Is this interface in use? [yes]: y
Configure IP on this interface? [yes]: y
IP address for this interface: 192.168.1.1
Number of bits in subnet field [0]: 0
Class C network is 192.168.1.0, 0 subnet bits;
mask is /24
```

There is a statement here in the Setup script that sometimes causes confusion. The question is the number of bits in the subnet **field**. This is not the same as the subnet mask! As shown above, the IP address for the interface is an address in a Class C network (192.168.1.0). The Setup script asks us for the number of bits in the **subnet field**. This means anything beyond the normal Class C mask of 24 bits or 255.255.255.0. If we were to decide that we wanted to subnet this network with a subnet mask of 255.255.255.192 (equivalent to 26 bits), we would respond to that question with the number 2.

For another example of this, we could say that we entered an IP address of 10.1.1.1, and we wanted it subnetted with the same size network that a Class C network has. This mask would be 24 bits, or 255.255.255.0. Since 10.1.1.1 is an address in a Class A network, the Setup script would consider the network mask to be 8 bits or 255.0.0.0. If we wanted the 24-bit mask, we would respond to this subnet field question with the answer 16. 8+16=24, and that gives us the 255.255.255.0 subnet mask. Subnet masking is covered in more detail in the CCNA Tutorial on IP Addressing. Now, back to the configuration:

```
Configuring interface Serial0:
Is this interface in use? [no]: y
Configure IP on this interface? [no]: y
Configure IP unnumbered on this interface? [no]: n
```

```
IP address for this interface: 192.168.2.1
Number of bits in subnet field [0]:
Class C network is 192.168.2.0, 0 subnet bits;
mask is /24
```

```
Configuring interface Serial1:
Is this interface in use? [yes]: n
```

The following configuration command script was created:

```
hostname Router
enable secret 5 $1$gTpr$wimCVlieyQAMEP/vfkEeF1
enable password cisco2
line vty 0 4
password cisco
no snmp-server
!
no appletalk routing
no decnet routing
ip routing
no clns routing
no ipx routing
no vines routing
no xns routing
no apollo routing
no bridge 1
!
interface Ethernet0
ip address 192.168.1.1 255.255.255.0
no mop enabled
!
interface Serial0
ip address 192.168.2.1 255.255.255.0
no mop enabled
!
interface Serial1
shutdown
no ip address
!
router rip
network 192.168.1.0
network 192.168.2.0
!
end
```

```
Use this configuration? [yes/no]: y
Building configuration...
Use the enabled mode 'configure' command
to modify this configuration.
```

Now we are done with the Setup script, and the router has entered the configuration commands to setup the router the way we specified. All that is left now is to reboot the router, and we are done. After you answer "yes" to the "Use this configuration?" question, a bunch of stuff will happen. This is okay. What is happening is that the router is doing a quick reset and implementing the configuration you just entered.

Press RETURN to get started!

```
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial1, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0,
changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1,
changed state to down
%LINK-5-CHANGED: Interface Serial0, changed state to down
%LINK-5-CHANGED: Interface Serial1, changed state to
administratively down
%SYS-5-RESTART: System restarted --
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
```

At this point, the router will appear to be hung up, but don't worry, it's just waiting for you to push it along. Press the Enter Key here and you will see the router prompt show up. If you followed the configuration above, notice the router prompt when you do hit the Enter key, it should look like this:

```
<cr>
Router>
```

Manual Router Configuration

Right after running the Setup script, or whenever you need to change your router's configuration, the best way to get the most out of a Cisco router configuration is to do it manually via the CLI and config mode.

Let's pretend that we didn't go through the Setup script as shown in the last section. Through our terminal emulation program, we will see the router boot up just like before. When we get the prompt asking if we'd like to enter the initial configuration dialog, we can simply type "n" as shown below and then press the Enter Key. The router will ask us if we want to terminate autoinstall. Our answer to this will be "yes." If we were to say "no," the router would spend a lot of time looking for configuration files from network servers. (We'll discuss this more in a later section.) As shown in the Setup script instructions, another way to get out of the Setup script is to just press [Ctrl-c] at any point in the script.

```
Notice: NVRAM invalid, possibly due to write erase.
--- System Configuration Dialog ---
```

```
At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[' ].
Would you like to enter the initial
configuration dialog? [yes]: n
```

```
Would you like to terminate autoinstall? [yes]: y
Press RETURN to get started!
```

```
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINK-3-UPDOWN: Interface Serial1, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0,
changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1,
changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
changed state to down
%LINK-5-CHANGED: Interface Ethernet0, changed state to
administratively down
%LINK-5-CHANGED: Interface Serial0, changed state to
administratively down
%LINK-5-CHANGED: Interface Serial1, changed state to
administratively down
%SYS-5-RESTART: System restarted --
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L), Version 11.2(18),
RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner
```

As you can see here, a couple things happen when you exit from the Setup script. Nothing that happens here is very complicated or problematic. This is a simple system reset that occurs as the router tries to boot. You'll notice above that the router tells you to Press RETURN to get started! This means that the router will wait for your input to present you with a prompt.

Once you exit the Setup script, you can manually configure the router. In order to configure the router via the CLI, you must do three things: first login to user mode, then login to privileged mode, and then enter configuration mode.

After exiting the Setup script (and pressing Enter), you will be presented the **Router>** prompt. This is the prompt of the user exec mode and it's like the DOS prompt on a PC. As described in an earlier section, user exec mode has limited functionality and no configuration access. From this prompt, you must enter the command **enable**, or in CLI shorthand, **en**. Once you have done this, you will be presented the **Router#** prompt (privileged exec mode prompt). Since we are looking at this as if there were no previous configuration on the router, we shouldn't be prompted for a password because currently there are no passwords associated with logging in or entering enable mode. We will cover that after we finish the initial configuration section.

Now that we have entered privileged mode, we can proceed to configuration mode, also known as config mode. From the Router# prompt, type in the command **configure terminal**, in shorthand, **conf t**, and press the Enter key. In config mode, you will see the Router(config)# prompt. From this config prompt, we can enter any configuration commands to setup the router to fit our needs. If you remember the Setup script that we went through, we will show the exact same configuration, but step by step as we go through the initial manual configuration.

In the previous configuration, we configured the router to do the following:

- Set router name
- Set passwords
- Configured interfaces and IP addressing (for Ethernet 0 and Serial 0 interfaces)

- Configured RIP routing for IP
- Committed the configuration to memory (NVRAM)

Now, we will do this via the CLI and router config mode. To begin, once again, we will login to the router, enter enable mode and then begin config mode. The following sequence will get us to this point:

```
<cr>
Router> en
Router# conf t
Router(config)#
```

Now that we are in config mode, we will run through the configuration items listed above. In each of these command sequences, you will see an abbreviated version of the command. If you'd like to see the full text version of the commands, remember that you can hit the [tab] key to fill in the rest of the command. In order to rename the host, you need to use the command host followed by the name you want for the router. In this case we picked "Router". To set the router name we follow this sequence:

```
Router(config)# host Router
Router(config)#
```

Setting Router Passwords

Security is important on a router. Remember that you can access a router from connections other than the console. There are five separate passwords you can set to protect your router:

Console: protects the Console Port

Enable Password: guards the use of the Enable mode super-user status

Enable Secret: an encrypted secret form of the above (better!)

VTY: protects against unauthorized Telnet port logons

Auxiliary: protects the AUX Port (for your modem)

The Console Password

As we continue with our manual reconfiguration tasks, your very next step should be to set the password for the Console Port. Starting from the Router(config)# prompt you need to put in the following series of commands to create the password.

```
Router(config)# line console 0
Router(config-line)# login
Router(config-line)# password cisco
Router(config-line)# Ctrl-Z
Router#
```

Notice that the Router prompt changes to Router(config-line) when you put in the **line console 0** command. It's important to know that **line** is a major command that puts you into "sub-command" mode. This is similar to another "sub-command" mode used for configuring interfaces Router(config-if). Only in the Router(config-line)# mode can you configure individual "lines." Also note that the Ctrl-Z (also written ^Z) ends your session, and brings you back up to the Router# prompt.

The Enable Password and Enable Secret Password

There are two different passwords that allow access to privileged mode, the enable password and the enable secret password.

The purpose of the enable password is to prevent unauthorized access to the privileged mode and configuration mode of the router. You can set this password from configuration mode like this:

```
Router(config)# enable password cisco2
```

The "enable secret" password is a "one-way cryptographic secret password." In other words, once you put in the plain text password, the Cisco IOS takes the text and encrypts it so that no one, not even you, can ever read it again. This is why it is good advice not to forget your enable secret password. Also, the router doesn't like the enable secret to be the same as the enable (as we'll see soon). You can set this password from configuration mode like this:

```
Router(config)# enable secret cisco
```

While you can configure both of these passwords, you can only use one of these passwords at a time on your router. If you choose the enable secret password (and you should because it uses an encryption algorithm to keep your password secure), the enable password will not be used. Even though you may have set your enable password and see it in your configuration file, if you also set the enable secret password your enable password will be ignored.

The only time you should rely only on the enable password is if you are working with an old version of the Cisco IOS (prior to version 10.3) or if your router has an older boot ROM that doesn't recognize the enable secret command. In other words, if your router is new, use the enable secret password.

If you entered the same password for enable secret as you did for enable password, you would receive the following message:

```
Router(config)#enable secret cisco
The enable secret you have chosen is the same as
your enable password. This is not recommended.
Re-enter the enable secret.
```

The IOS gives you this warning because your enable password is listed in clear text right in your configuration file, which anyone can see from user exec mode. If your enable secret password is the same as your enable password and your enable password is shown in your configuration file, you've essentially given away your enable secret password to anyone who guesses that they may be the same.

The VTY Password

VTY ports are not real ports. In other words, you won't find a port on the back of your router labeled VTY. They are also called "Virtual Ports" and they wait for a remote connection, usually using Telnet, to log in. So the virtual terminal password is essentially the same as a Telnet password. Configuring the VTY password is very similar to configuring the Console. The only difference is that there are five VTY virtual ports, which are named 0, 1, 2, 3, and 4. You can use the shortcut 0 4 (a zero, a space, and 4) to set all five passwords at the same time. In order to set the virtual terminal (vty) password, you must enter the following configuration commands:

```
Router(config)# line vty 0 4
Router(config-line)# pass cisco
Router(config-line)# exit
Router(config)#
```

Notice we see the config-line text after the router prompt. When you see this, the commands you are entering here are specific to a certain line interface. If you are in an interface or line configuration mode, you can get back out to the global config mode by typing in **exit** and pressing Enter instead of using CTRL-Z (^Z). By the way, it is not necessary to exit back to the global config mode every time you are finished configuring one interface. You can travel from one interface to another by giving the next interface command.

The Auxiliary Line Password

You can set the Auxiliary Line Password for external modem connections by entering the following commands:

```
Router# config t
Router(config)# line aux 0
Router(config-line)# login
Router(config-line)# password cisco3
Router(config-line)# Ctrl-Z
Router#
```

And now your Router has a password protecting the AUX port.

Now that you have successfully entered all the passwords your router needs, this is a good time to do a quick practice session. To leave the enable mode you need to type in the word **disable** (the opposite of **enable**, the command that got you into this mode). Remember again that enable mode is formally called "Privileged Exec Mode."

```
Router# disable
```

This will leave you at the User Exec Mode prompt, Router >. Now you are going to leave User Exec Mode by typing **quit** or **exit**:

```
Router> exit (or type quit)
```

You will now see the friendly message:

```
Press ENTER to get started.
```

At this point press the ENTER key. The next thing you will see on the screen will be:

```
User Access Verification Password
(please type in your User Password here)
```

```
Router>
```

You quickly recognize the "Router >" as the User Exec Mode prompt. Now type in your Enable Secret Password.

```
Router> cisco2
```

If you typed in your enable Secret Password correctly, you should now be in the Privileged Exec Mode.

```
Router#
```

Congratulations! You have now set up your router, created passwords, and successfully logged back into it. Now don't forget your passwords!

Configuring an IP Address

As we continue with the manual reconfiguration of our router, we will now configure two interfaces: Ethernet0 and Serial0. To configure IP routing and then configure Ethernet0 and Serial0 interfaces with IP addresses we enter the following:

```
Router(config)# ip routing
Router(config)# int e0
Router(config-if)# ip add 192.168.1.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# int s0
Router(config-if)# ip add 192.168.2.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)#
```

You can see that we entered commands for the Ethernet0 and Serial0 interfaces without having to exit back to the global config mode between interface commands.

Notice that we are in the global config mode when we turn on IP routing. However, turning on IP routing is not necessary on any Cisco router, because IP routing is enabled by default.

We enter the interface "subcommand" mode with `int e0` (interface Ethernet0). We must specify the subnet mask when entering an IP address. Notice that we are using the full sense of the 24-bit mask with three 255s, where each 255 represents 8 bits. Remember we entered a zero when we ran the Setup script, because the Setup script uses zero to represent subnet mask defaults, and in this case, the default subnet mask for this Class C network is 24 bits.

Notice the use of the command **no shutdown**. Use this command to change the state of the interface to up. Remember that an interface is always "shut down" by default and it is a good idea to enter no shutdown just to be on the safe side. If your Ethernet 0 interface is plugged in correctly and you enter a no shutdown, you will see the following appear on the screen that tells you that your newly configured Ethernet0 interface is working:

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
                        changed state to up
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
```

Since we enabled IP routing, we must turn on some type of routing protocol for IP route tables to propagate through the network. One such routing protocol is RIP, Routing Information Protocol. To enable RIP for IP, we enter:

```
Router(config)# router rip
Router(config-router)# network 192.168.1.0
Router(config-router)# network 192.168.2.0
Router(config-router)#
```

In the above example, RIP is advertising two networks, 192.168.2.0 and 192.168.1.0, so other connected routers can build routing tables.

Note: Now that we finished reconfiguring the most important parts of the router's configuration, let's learn about other configuration features that are either necessary for router maintenance or act as enhancements for day-to-day operations.

Configuring Banners

An IOS banner is used to give information to users or administrators when they log in to a router via a terminal line. We are going to cover three types of banners:

- MOTD (Message Of The Day)
- Login
- Exec

An MOTD banner is sent to a terminal as soon as the terminal's connection becomes active. A login banner is also sent to a terminal when a terminal becomes operative. The login banner is displayed after an MOTD banner if there is one. An exec banner is displayed to a terminal immediately after a person has successfully logged in. We use the global configuration mode command **banner** to create a banner.

```
Router# banner {exec | login | motd} dc message dc
```

The argument **dc** is a delimiting character. The delimiting character can be any character as long as it is not part of the message, and it must be the same at the end as it is at the beginning of the **message**. The banner command should include one of the arguments **exec**, **login**, or **motd**. All three types of banners can be created by issuing the banner command three times, once for each type.

To create a banner, type the command including the first delimiting character, and press Enter. On the next blank line type the rest of the banner message. Banners can have multiple lines (that's the reason for using a dc). When you have finished with the banner message, just type the delimiting character and press Enter again. In the following three examples we will use the percent sign (%) as the delimiting character. Everything between the percent signs is the banner. When the configuration is displayed, IOS uses its own standard delimiter (^C).

```
Router(config)# banner motd %
Enter TEXT message. End with the character '%'.
This is the motd banner.
Remember to meet in cafeteria for Norman's party.
%
Router(config)# banner login %
Enter TEXT message. End with the character '%'.
This is the login banner.
You have accessed a private system.
Unauthorized access is prohibited.
%
Router(config)# banner exec %
Enter TEXT message. End with the character '%'.
This is the exec banner.
We just added a new IOS to all routers.
%
Router(config)#
```

Committing Configuration Changes to NVRAM

All configurations entered through the configuration mode are done dynamically, and stored in regular RAM (volatile RAM) as the current running configuration. Now that we are done with the configuration, we must exit config mode and save our changes. To exit configuration mode you can either enter the command **end**, or press the [Ctrl-z] key combination.

In order to commit these changes to non-volatile RAM (NVRAM) so that they will be enabled every time the router is restarted, we must "write" this running configuration to memory. Remember that the saved configuration in NVRAM is called the "Startup Configuration." Previous to IOS version 10.3, there was only one command to enter, **write memory**, or **wr mem**. In IOS versions 10.3 and later we can use either the old command **wr mem** or the newer command **copy running-config startup-config**, with the shorthand version being **copy run start**. This newer command is preferred by Cisco and is the one you will see most often in textbooks, courses, and exams.

Here are two ways you can exit configuration mode and commit your changes to NVRAM:

```
Router(config)# end
Router# wr mem
```

or

```
Router(config)# ^Z (just pressed [ctrl-z])
Router# copy run start
```

When you commit the running config to memory on a 2500 series router, you will have a little pause while the router saves the config into NVRAM. When it is done, the router will respond to you with the congenial message

```
Building configuration. . .[OK].
```

To summarize the whole configuration we have just gone through, here are all the commands to accomplish the above without interruptions:

```
<cr>
Router> en
Router# conf t
Router(config)# host Router
Router(config)# ena sec cisco
Router(config)# ena pass cisco2
Router(config)# line vty 0 4
Router(config-line)# pass cisco
Router(config-line)# exit
Router(config)#
Router(config)# ip routing
Router(config)# int e0
Router(config-if)# ip add 192.168.1.1 255.255.255.0
```

Notice the from/to syntax of the copy command. This is a critical concept to understand, and there can be serious repercussions if the command is written incorrectly.

With **copy run start** you are telling the IOS that you want to **copy from** the **running-config** to the **startup-config**. This will effectively save the configuration you've been changing to the more permanent startup-config in NVRAM.

However, if you were to type **copy start run**, you are telling the IOS that you want to **copy from** the **startup-config** to the **running-config**. This **restores** your running-config to the way it looked when your router booted (or to the most recently saved startup configuration) erasing any changes you've made to your running-config.

So remember, with the copy command the syntax is always from/to!

Later in the paper we'll discuss other examples of the copy command where the from/to syntax still applies.

```

Router(config-if)# no shut
Router(config-if)# int s0
Router(config-if)# ip add 192.168.2.1 255.255.255.0
Router(config-if)# no shut
Router(config-if)# router rip
Router(config-router)# network 192.168.1.0
Router(config-router)# network 192.168.2.0
Router(config)# ^Z (just pressed [ctrl-z])
Router# copy run start
Building configuration...
[OK]

```

Now that we are done with creating the configuration and saved it, we can view it by entering the command **show running-config** or **sh run**. Here's what it will look like:

```

Router# sh run
Building configuration...

Current configuration:
!
version 11.2
no service password-encryption
no service udp-small-servers
no service tcp-small-servers
!
hostname Router
!
enable secret 5 $1$r.0I$4MbN8jBZLXq9siy9R1ELR1
enable password cisco2
!
!
interface Ethernet0
 ip address 192.168.1.1 255.255.255.0
!
interface Serial0
 ip address 192.168.2.1 255.255.255.0
 no fair-queue
!
interface Serial1
 no ip address
 shutdown
!
router rip
 network 192.168.1.0
 network 192.168.2.0
!
no ip classless
!
!
line con 0
line aux 0
line vty 0 4
 password cisco
 login
!
end

```

If your configuration looks like this, GOOD JOB!

Configuring Clock Rate

We'll end this chapter on Basic Router Configuration with a brief description of what clock rate is and how to configure it.

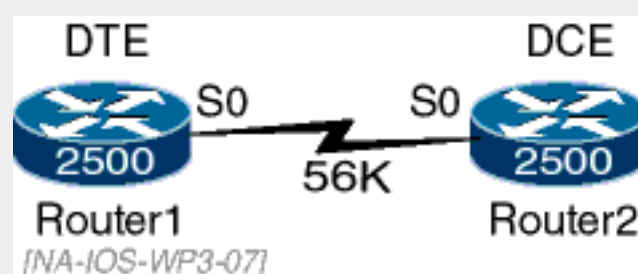


Figure 7. Diagram of Two Routers Connected to Each Other

Clock rate is the command you enter to supply the clock signal that paces the communications on a circuit. For instance, imagine that we wanted to connect two routers together as in Figure 7. In this case, we will connect the routers together with a serial cable (for more information on making connections with serial cables, see "[Let's Connect: Your Serial Cable Guide](#)". (is not associated with Cisco.)

If we wanted this link to simulate a 56K circuit, we could enter the following commands in Router2:

```
Router2# config t
Router2(config)# int s0
Router2(config-if)# clock rate 56000
Router2(config-if)# [ctrl-z]
```

Because Router2 is providing the clock for the circuit between itself and Router1, it is known as the Data Communications Equipment (or Data Circuit-terminating Equipment (DCE) depending on how you want to think about it). Router1 accepts this clock rate from Router2, which makes Router1 the Data Terminal Equipment (DTE).

Basic Router Maintenance and Troubleshooting

Backing up Router Configuration Files

There are several ways to back up the configuration files you've worked so hard to create. The safest and most common way to do these backups is by using a Trivial File Transfer Protocol (TFTP) server. (This is different from the FTP protocol.)

The TFTP server serves as a central configuration repository. This one location can store the router configurations of all the routers in your network. This can be handy in the case of a router failure that causes it to lose its configuration or even if the router itself breaks and you have to get a replacement from Cisco. In these instances, it is possible to retrieve the configurations from a TFTP server and quickly have your router running with your most recently saved configuration.

A TFTP server can run on almost any computer with nearly every operating system. If you are interested in trying out a simple TFTP server for learning purposes, you can [get one from the Cisco web site](#) without having to log in. (is not associated with Cisco.) This TFTP server will run on Windows 95/98/NT and is very simple to install and setup. Of course there are other shareware TFTP server programs on the Web that you can try.

In order to save the configuration of your router to the TFTP server, you must first have some sort of network connectivity that will allow you to get to the TFTP server. The computer with the TFTP server running on it needs to be on the same Ethernet segment as your router. The easiest way to accomplish this is to have them both connected to a hub. This means that your router's AUI port will need an Ethernet transceiver with an RJ45 port. With this in place, you can save and retrieve configuration files to and from the TFTP server.

If the TFTP server IP address is 192.168.1.100, the procedure to save the configuration file to the TFTP server is as shown below. We first want to ping the IP address of the TFTP server to ensure that there is connectivity between the router and the server, and then we use the command **copy running-configuration tftp** or **copy run tftp**.

```
Router# ping 192.168.1.100
Type escape sequence to abort.
Sending 5 100-byte ICMP echos to 192.168.1.100,
  timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5),
  round-trip min/avg/max = 4/4/4 ms
Router# copy run tftp
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file{}? My-config <cr>
```

Now here's what the commands look like if you wanted to retrieve your configuration from the TFTP server to your router.

```
Router# copy tftp run
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file{}? My-config <cr>
Configure using My-config from 10.1.1.1? [confirm] <cr>
Loading My-config...from 10.1.1.1 (via Ethernet0):
[OK - 717/32732]
Router#
```

Building configuration...

This just copies a saved configuration from the TFTP server to your currently running configuration, not your startup configuration. You can either do a **copy run start** to save this configuration to your startup configuration or you can load a saved configuration copy from the TFTP server directly to your startup configuration located in NVRAM. However, in order for this configuration to become your active configuration, you would then have to

Probably one of the most confusing things about building a network is trying to figure out how everything connects together. While this is primarily a physical cabling issue outside the scope of this paper, it is important to understand that the way you physically connect your equipment can affect the way you need to configure your routers.

Devices that communicate over a serial interface are either DCE or DTE. Some devices are always DCE like modems, CSU/DSUs, and multiplexers. Data terminals are always DTE (thus the name). However, routers, hubs, and switches can be either DCE or DTE.

Believe it or not, the end of the serial cable that's plugged into the router determines whether it is DCE or DTE. Generally, the female end of the serial cable makes the router the DCE. Likewise, the male end of the serial cable makes the router the DTE.

If the router is the DCE, it needs to set the clock rate for the circuit. This shows that physical connections can sometimes impact your router configuration.

For a more in-depth explanation of what serial cables to use in what circumstances, see "[Serial Cables](#)." (is not associated with Cisco.)

In addition to storing router configurations on a central server, you can also use a core router in your network as a TFTP server. For example, consider a typical hub-and-spoke network configuration that has a 7500 series router at the hub. On this central router you could outfit the router processor card with a decent size PCMCIA Flash card and use this as a storage compartment for all remote site router configuration files. Also, when each router boots each could be configured to retrieve its configuration from that 7500 series router, acting as a TFTP server.

copy the startup configuration to the running configuration by reloading your router.

```
Router# copy tftp start
Host or network configuration file [host]? <cr>
Address of remote host [255.255.255.255]? 10.1.1.1
Name of configuration file [My-config]? <cr>
Configure using My-config from 10.1.1.1? [confirm] <cr>
Loading My-config ...
```

Instead of **reload**, you could use the command **copy start run** without requiring a reset. The problem with **copy start run** is that it actually merges the two configuration files and doesn't make a clean copy. This is not recommended unless you're sure you know what you're doing.

Another alternative to the **reload** command is to use the command **conf mem** (short for **configure memory**), which executes the commands stored in NVRAM.

Now you need to put the saved configuration in startup back into the running configuration by resetting the router. The command to do this is **reload**, and it's this simple.

```
Router# reload
```

And now you know two methods of backing up your router's configuration files. Why would you want to do this? Well, it is good for resetting the router back to square one if you make a mistake. It is also good for doing a practice Lab a second time.

Which brings us to the ultimate of all configuration commands: **erase startup-config**. This command erases your NVRAM so that the next time you reload, you have a completely blank router. You can use this command to practice the Setup Script covered in an earlier section. Do **not** use this on a production router, as this will bring down your network and likely your job.

Backing Up IOS Software Images

You can also save your router's Flash memory, where the Cisco IOS is stored, to a TFTP server. The following shows the commands to copy Flash to the TFTP server. Notice that you are asked for the IP address of the TFTP server and the name of the file that contains the Cisco IOS on your router. You can find the name of this file by using the **show flash** command in the global configuration mode.

```
Router# copy flash tftp
File Length name/status
11233404 c2500-ajs40-1_113-5_T.bin
[11233468 bytes used, 5543748 available, 16777216 total]
Address or name of remote host [255.255.255.255]? 10.1.1.1
Source file name? c2500-ajs40-1_113-5_T.bin
Destination file name[c2500-ajs40-1_113-5_T.bin]? <cr>
```

You can do it the other way and copy Flash from the server to the router.

```
Router# copy tftp flash
**** NOTICE ****
Flash load helper v1.0
This process will accept the copy
options and then terminate the current
system image to use the ROM based image
for the copy. Routing functionality will
not be available during that time. If
you are logged in via Telnet, this
connection will terminate. Users with
console access can see the results of
the copy operation.
---- ***** ----
Proceed? [confirm] <cr>

System Flash directory:
File Length Name/status
 1 8121000 c2500-js-1.112-18.bin
[8121064 bytes used, 8656152 available, 16777216 total]
Address or name of
remote host [255.255.255.255]? 192.168.1.100
Source file name? c2500-js-1.112-18.bin
Destination file name [c2500-js-1.112-18.bin]? <cr>
Accessing file 'c2500-js-1.112-18.bin' on 192.168.1.100...
Loading c2500-js-1.112-18.bin
from 192.168.1.100 (via Ethernet0): ! [OK]
```

Another way you can back up your router configuration files is to save them to your router's Flash memory. Configuration files are relatively small and there is usually plenty of Flash space on a new router. To see how much Flash space your router has and the names and number of files located there, type **show flash** in the global configuration mode.

You can save your configurations to Flash by typing **copy run flash** or **copy start flash** depending on which configuration files you wish to save. The IOS will prompt you to name the file you are about to save.

While saving your configuration files to Flash is an option if you don't have a TFTP server, a TFTP server is a safer backup method. Since it is located on a device other than your router it is still available in case something catastrophic happens to your router. Also, the files stored on a TFTP server can be included in regular network backups.

```

Erase Flash device before writing? [confirm] <cr>
Flash contains files. Are you sure you want
  to erase? [confirm] <cr>

Copy 'c2500-js-1.112-18.bin' from server
  as 'c2500-js-1.112-18.bin' into Flash
  WITH erase? [yes/no] y

00:10:59: %SYS-5-RELOAD: Reload requested
%SYS-4-CONFIG_NEWER: Configurations from
  version 11.3 may not be correctly understood.
%FLH: c2500-js-1.112-18.bin
  from 192.168.1.100 to Flash ...

System Flash directory:
File Length Name/status
  1 8121000 c2500-js-1.112-18.bin
[8121064 bytes used, 8656152 available, 16777216 total]
Accessing file 'c2500-js-1.112-18.bin' on 192.168.1.100...
Loading c2500-js-1.112-18.bin
  from 192.168.1.100 (via Ethernet0): ! [OK]

Erasing device... eeeeeeeeeeeeeeeeeeee ...erased
Loading c2500-js-1.112-18.bin
  from 192.168.1.100 (via Ethernet0):
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!
(There may be many more or fewer of
  these ! marks on your screen.)

[OK - 8121000/16777216 bytes]

Verifying checksum... OK (0x5996)
Flash copy took 0:04:10 [hh:mm:ss]
%FLH: Re-booting system after download
(Here the router reboots like normal.)

```

You may have noticed that the router had to reboot in order to copy the image into Flash. This is okay, but we should investigate the reason why this happens.

Because this is a 2500 series router that runs the IOS straight from Flash, copying a file to Flash while the IOS is running from Flash will not work. In order to make software copies and updating as painless as possible, Cisco created the Flash Load Helper. The Flash Load Helper assists in setting the configuration register to run from ROM, reboots the router, copies the image into Flash, changes the configuration register back, then reboots the router again.

The Flash Load Helper was an optional feature in Cisco routers until software version 10.3. If you happen to be working with a router with earlier software, you must manually set the configuration register to boot with the default software in ROM, then perform the image copy, then change the configuration register back and reboot the router. We'll discuss the configuration register later in the paper. Most other router models run the IOS from RAM, so this may not be an issue if you are using another router model.

Show Commands

While making changes to your router's configuration, it is a good idea to frequently monitor your work before going on. The privileged mode is where we can examine our work by using certain verification (show) commands. After performing several manual configuration tasks either in the global configuration mode or other deeper "subcommand" configuration modes, you should slip back into the privileged mode to check things out.

If you're at the Router# prompt, just type **exit**, and that will send you back to the Router> prompt, which is where you need to be to use the verification commands.

We will examine the following verification commands and the information they provide. The **show** command is the portal that allows us to view anything we want on the router. If you'd like to see what show options are available, type **show ?** at the exec prompt.

We will discuss the more common show commands.

It's useful to know older commands from Cisco IOS version 10.3 or lower that are equivalent to those in recent software versions. Table 2 displays equivalent commands between newer and older software version.

Table 2. Equivalent Commands

| Higher than Version 10.3 | Version 10.3 or lower |
|--------------------------|-----------------------|
|--------------------------|-----------------------|

| | |
|------------------------------------|--------------------|
| Show startup-configuration | Show configuration |
| Show running-configuration | Write term |
| Erase startup-configuration | Write erase |
| Copy running-config startup-config | Write mem |

Most show commands can be viewed from the regular User Exec mode. Some show commands can only be viewed from the Privileged Exec (Enable) mode. If you've been busily configuring interfaces and protocols in config mode and forget to change back to the Router# or Router> prompt, using a show command will not work. None of the show commands can be used from config mode. This will just give you an error, and you will feel very silly.

If you type in the command **show**, a space, and then a question mark at the Enable Mode "Router#" prompt, the Help function will give you a long list of show commands.

```
Router# show ?
show access-expression show access-list
show apple interface
show apple route
show appletalk
show atm
show bridge
show cam
show cam dynamic
show cdp neighbors
show config
...
```

...and so on going down through the entire alphabet. Luckily, you do not need to memorize all these right away for the tests. There are, however, two important show commands that allow you to see your router's full configurations and you'll want to remember: **show running-configuration** and **show startup-configuration**.

show startup-config shows you the configuration commands stored in the Router's NVRAM, the place where configurations live when the power is off.

The **show running-config** command shows you the configuration as you have changed it since turning on the router. We used this command earlier in the paper to show the changes we made to the router. This command shows the configuration that is actually running right now on your router, in RAM.

For security reasons, these commands are not available from the user prompt. If you do a **show run** or **show start** from the Router> prompt, you'll get an error message. The reason for this is that the enable password (but not the enable secret password) is shown in clear text by these commands.

Basic Show Commands

You are apt to use most or all of the information in these commands when doing routine troubleshooting.

Show Version

The **show version** command gives you information on the version of the Cisco Internetwork Operating System that your router is using. It also gives you lots of other basic information such as how long the router has been up, how the system was started, what processor your router uses, how much memory your router has, and from where the system image file was loaded. **show version** will also show you what interfaces the router has.

```
router# show version
Cisco Internetwork Operating System Software IOS (tm)
3000 software (IGS-I-L, Version 11.1(11)
  RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by Cisco Systems, Inc.
Compiled Tue 24-Jun-97 12:20 by jaturner
Image text-base: 0x0301E644, data-base 0x00001000

ROM: System Bootstrap, Version 11.0(10c), SOFTWARE ROM:
  3000 Bootstrap Software (IGS-BOOT-R),
  Version 11.0(10c) RELEASE SOFTWARE (fc1)

Router uptime is 12 minutes System restarted by power-on
System image file is "Flash:igs-i-l.110-16",
  booted via Flash

cisco 2500 (68030) processor (revision N) with
  2048K/2048K bytes of memory.
Processor board ID 06267777, with hardware revision 00000000
```

Bridging software X.25 software, Version 2.0, NET2,,
BFE and GOSIP compliant.

1 Ethernet/IEEE 802.3 interface. 2 Serial network interfaces.

32K bytes of non-volatile configuration memory.
8192K bytes of processor board System Flash (Read ONLY)

Configuration register is 0x2102

Show Interfaces

The **show interfaces** command is like the Swiss Army knife of troubleshooting. It gives you information on all the interfaces in your router. Since the interfaces are where all the real work takes place, being able to see what they are doing is very helpful. In the example below, we are using a Cisco Router 2503, which includes two additional ISDN ports (BRI0:1 and BRI0:2).

One of the most important ways to check the health of a Cisco router interface is on the first line of this output. In this case, the serial interface is administratively down, and the line protocol is down. This means that the interface is in "shutdown" mode, and whoever is configuring this router wants that interface to remain shut down. If that line were to say that the interface was down instead of administratively down, that means that we have a problem with the physical connectivity on that line. Another basic status we could get on this interface is that the interface is up, but the line protocol is down. A lot of times this means that the OSI model Layer 1 is up but Layer 2 is down. Yet another status is that the line is up and the line protocol is up. This is how it should be, and it means everything is okay on that interface.

Router> **show interfaces**

```
BRI0 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
    Conversations 0/0/256 (active/max active/max total)
    Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
    0 packets input, 0 bytes, 0 no buffer
    Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
    0 input errors, 0 CRC, 0 frame, 0 overrun,
    0 ignored, 0 abort
    0 packets output, 0 bytes, 0 underruns
    0 output errors, 0 collisions, 5 interface resets
    0 output buffer failures, 0 output buffers swapped out
    0 carrier transitions
```

```
BRI0:1 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
    Conversations 0/0/256 (active/max active/max total)
    Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
    0 packets input, 0 bytes, 0 no buffer
    Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
    0 input errors, 0 CRC, 0 frame, 0 overrun,
    0 ignored, 0 abort
    0 packets output, 0 bytes, 0 underruns
    0 output errors, 0 collisions, 5 interface resets
    0 output buffer failures, 0 output buffers swapped out
    0 carrier transitions
```

```
BRI0:2 is administratively down, line protocol is down
Hardware is BRI
MTU 1500 bytes, BW 64 Kbit, DLY 20000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
```

```
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
  Conversations 0/0/256 (active/max active/max total)
  Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
  0 input errors, 0 CRC, 0 frame, 0 overrun,
    0 ignored, 0 abort
  0 packets output, 0 bytes, 0 underruns
  0 output errors, 0 collisions, 5 interface resets
  0 output buffer failures, 0 output buffers swapped out
  0 carrier transitions
```

```
Ethernet0 is administratively down, line protocol is down
Hardware is Lance,
address is 0010.7b3a.dea6 (bia 0010.7b3a.dea6)
MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
  reliability 252/255, txload 1/255, rxload 1/255
Encapsulation ARPA, loopback not set
Keepalive set (10 sec)
ARP type: ARPA, ARP Timeout 04:00:00
Last input never, output 01:17:16, output hang never
Last clearing of "show interface" counters never
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
  0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
  0 input packets with dribble condition detected
  14 packets output, 840 bytes, 0 underruns
  14 output errors, 0 collisions, 1 interface resets
  0 babbles, 0 late collision, 0 deferred
  14 lost carrier, 0 no carrier
  0 output buffer failures, 0 output buffers swapped out
```

```
Serial0 is administratively down, line protocol is down
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output 01:17:18, output hang never
Last clearing of "show interface" counters 01:17:18
Queueing strategy: fifo
Output queue 0/40, 0 drops; input queue 0/75, 0 drops
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
  0 packets input, 0 bytes, 0 no buffer
  Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
  0 input errors, 0 CRC, 0 frame, 0 overrun,
    0 ignored, 0 abort
  5 packets output, 853 bytes, 0 underruns
  0 output errors, 0 collisions, 2 interface resets
  0 output buffer failures, 0 output buffers swapped out
  0 carrier transitions
DCD=down DSR=down DTR=down RTS=down CTS=down
```

```
Serial1 is administratively down, line protocol is down
Hardware is HD64570
MTU 1500 bytes, BW 1544 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output 01:17:50, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
  Conversations 0/2/256 (active/max active/max total)
  Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
```

```

5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun,
 0 ignored, 0 abort
6 packets output, 132 bytes, 0 underruns
0 output errors, 0 collisions, 3 interface resets
0 output buffer failures, 0 output buffers swapped out
23 carrier transitions
DCD=down DSR=down DTR=down RTS=down CTS=down

Serial2 is administratively down, line protocol is down
Hardware is CD2430 in sync mode
MTU 1500 bytes, BW 115 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
  Conversations 0/0/256 (active/max active/max total)
  Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun,
 0 ignored, 0 abort
6 packets output, 1992 bytes, 0 underruns
0 output errors, 0 collisions, 2 interface resets
0 output buffer failures, 0 output buffers swapped out
0 carrier transitions
DCD=down DSR=down DTR=down RTS=down CTS=down

Serial3 is administratively down, line protocol is down
Hardware is CD2430 in sync mode
MTU 1500 bytes, BW 115 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input never, output never, output hang never
Last clearing of "show interface" counters never
Input queue: 0/75/0 (size/max/drops); Total output drops: 0
Queueing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
  Conversations 0/0/256 (active/max active/max total)
  Reserved Conversations 0/0 (allocated/max allocated)
5 minute input rate 0 bits/sec, 0 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
 0 packets input, 0 bytes, 0 no buffer
Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
0 input errors, 0 CRC, 0 frame, 0 overrun,
 0 ignored, 0 abort
6 packets output, 1992 bytes, 0 underruns
0 output errors, 0 collisions, 2 interface resets
0 output buffer failures, 0 output buffers swapped out
0 carrier transitions
DCD=down DSR=down DTR=down RTS=down CTS=down

```

Show Protocols

A protocol is an agreed-upon set of rules for speaking to others. It's sort of like having a conference call and everyone agreeing to speak German. The **show protocols** command lets you know if everyone is speaking properly. If they are not, then the router will tell you, "Line Protocol is down." Even if the interface is up, if the line protocol isn't working, nothing works.

All of our interfaces will be listed as administratively down since we have not yet turned any of them on. In fact, since we are only doing the basic setup of one router in this tutorial, we don't actually have anyone else with whom to speak.

```
Router> show protocols
```

```

Global values:
Internet Protocol routing is enabled
BRI0 is administratively down, line protocol is down
BRI0:1 is administratively down, line protocol is down
BRI0:2 is administratively down, line protocol is down

```

Ethernet0 is administratively down, line protocol is down
Serial0 is administratively down, line protocol is down
Serial1 is administratively down, line protocol is down
Serial2 is administratively down, line protocol is down
Serial3 is administratively down, line protocol is down

Show Flash

The **show flash** command tells you how many bytes are used and available in Flash memory and what files are stored there.

```
Router> show flash
System Flash directory:
File Length Name/status
  1 11780820 12-04T.bin
[11780884 bytes used, 4996332 available, 16777216 total]
16384K bytes of processor board System Flash (Read ONLY)
```

Advanced Show Commands

These commands do give some information that may be useful on a day-to-day basis, but much of the information they show is meaningful only to Cisco technical support. Technical support has access to router code, detailed internal data structures, and other information not provided to customers. This sort of confidential information is needed to understand the full meaning of advanced displays.

Show Memory

The **show memory** command shows what memory is allocated by the management system for which purposes. There are two memory charts that get shown by the command: a Summary and a Detailed Block by Block memory chart.

```
Router> show memory
```

Summary:

```
Head Total(b) Used(b) Free(b) Lowest(b) Largest(b) Processor
EA90C 5326580 2056220 3270360 3270360 3231192
I/O 600000 2097152 465264 1631888 1579032 1631720
```

A Detailed Block-by-Block memory chart:

```
Allocator PC Summary for: Processor
```

```
pc=0x031FDE54, size=000963416, count=000056,
  name=List Elements pc=0x031D8060, size=000462508,
  count=000312, name=*Packet Data*
pc=0x03217BAE, size=000287992, count=000068, name=Interrupt Stack
pc=0x031D8028, size=000178496, count=000312, name=*Packet Header*
pc=0x031DCDEC, size=000115040, count=000008, name=Fair Queueing
pc=0x031C2BD2, size=000049196, count=000001, name=Exec
pc=0x031DDBA8, size=000044660, count=000011, name=*Hardware IDB*
pc=0x031957E4, size=000040840, count=000010, name=TTY data
pc=0x03214150, size=000033516, count=000063, name=Process
pc=0x0322E6F4, size=000032808, count=000001, name=Cfg EEPROM Copy
pc=0x031DDBBE, size=000025124, count=000011, name=*Software IDB*
pc=0x034A829A, size=000014468, count=000001, name=Init
pc=0x034A81F4, size=000014464, count=000001, name=Init
pc=0x03AA68C2, size=000013644, count=000001, name=Init
pc=0x03A772B6, size=000013644, count=000028, name=ATMSIG-SHOW
pc=0x031A2D10, size=000013512, count=000197, name=Parser
01:13:41: %SYS-3-CPUHOG: Task ran for 2008 msec (19/19),
  process = Exec, PC = 31 7A068.

-Traceback= 320F2A6 317A070 318F4A4 31904A2 318F54C 31C2EBE
 31C3028 31C3332 31A18F0 31B605C Linkage
pc=0x031368E0, size=000012044, count=000001, name=Init
pc=0x0320BCD8, size=000012032, count=000084, name=Watched Boolean
pc=0x032B17D0, size=000011420, count=000001,
  name=DHCPD Message Workspace
pc=0x0320BEE8, size=000011040, count=000064, name=Process Events
--More--
```

Right now, we are only concerned with a few of the columns in this output. The Total, Used and Free columns give us a sense of how much operating memory the router has, how much is being used right now by the processor and the I/O subsystem, and how much is free to each. The (b) on each of these columns means that this information is expressed in bytes.

If you are interested in seeing how much memory each process running in the router is taking, you may want to look at the rest of the output from the **sh mem** command. We must warn you, this is a lot of information, and gets somewhat boring to look at.

Show Processes

A process is part of a program, or if it is small, it can be the entire program. It's sort of like having a troupe of jugglers: each item they toss up in the

air is one process. As long as they keep them all going, everything is fine. If not, you can use **show processes** to do a little troubleshooting. The show processes command shows you all the active processes in the form of a chart containing the following information in columns:

PID - The ID number of each process.

Q - The Queue priority

TY - This is the status of the process

PC - Program Counter.

Runtime - The amount of CPU time in milliseconds used by the process

Invoked - This is the amount of time the process has been invoked.

uSecs - The CPU time in milliseconds for each process invocation.

Stacks - This shows both the "low water mark" / "total stack space" in bytes.

TTY - Shows you which terminal controls the process.

Process - Finally, this actually gives you the name of the process.

Pay particular attention to the first line, which shows CPU utilization. While there are no hard and fast rules, you generally don't want a router that does dynamic routing to average much more than 50-60% of utilization over 5-minutes.

Router> **show processes**

CPU utilization for five seconds: 7%/7%;
one minute: 9%; five minutes: 12%

| PID | QTy | PC | Runtime (ms) | Invoked | uSecs | Stacks | TTY | Process |
|-----|-----|---------|-----------------|---------|-------|-----------|-----|-------------------|
| 1 | Csp | 32134FE | 8 | 872 | 9 | 736/1000 | 0 | Load Meter |
| 2 | M* | 0 | 3632 | 82 | 44292 | 2960/4000 | 0 | Exec |
| 3 | Lst | 3203DC6 | 14300 | 960 | 14895 | 3736/4000 | 0 | Check heaps |
| 4 | Cwe | 3209FB6 | 0 | 1 | 0 | 3724/4000 | 0 | Pool Manager |
| 5 | Mst | 318E706 | 0 | 2 | 0 | 3700/4000 | 0 | Timers |
| 6 | Mwe | 311F992 | 8 | 2 | 4000 | 3696/4000 | 0 | Serial Background |
| 7 | Lwe | 323C858 | 340 | 78 | 4358 | 3684/4000 | 0 | ARP Input |
| 8 | Mwe | 33877A6 | 0 | 3 | 0 | 3704/4000 | 0 | DDR Timers |
| 9 | Mwe | 339B8CA | 0 | 2 | 0 | 5712/6000 | 0 | Dialer event |
| 10 | Lwe | 34BE0AC | 36 | 2 | 18000 | 3684/4000 | 0 | Entity MIB API |
| 11 | Mwe | 3125CA2 | 0 | 1 | 0 | 3732/4000 | 0 | SERIAL A'detect |
| 12 | Cwe | 320D770 | 0 | 1 | 0 | 3740/4000 | 0 | Critical Bkgnd |
| 13 | Mwe | 31E55AA | 696 | 547 | 1272 | 4756/6000 | 0 | Net Background |
| 14 | Lwe | 31857B2 | 16 | 7 | 2285 | 5604/6000 | 0 | Logger |
| 15 | Msp | 319E1D4 | 172 | 4347 | 39 | 3568/4000 | 0 | TTY Background |
| 16 | Msp | 31E4EB6 | 3084 | 4415 | 698 | 3736/4000 | 0 | Per-Second Jobs |
| 17 | Msi | 3235488 | 40 | 4351 | 9 | 3724/4000 | 0 | Partition Check |
| 18 | Hwe | 31E5014 | 0 | 1 | 0 | 3712/4000 | 0 | Net Input |
| 19 | Csp | 31EC442 | 68 | 873 | 77 | 3728/4000 | 0 | Compute load avg |
| 20 | Msp | 31E4EE4 | 4740 | 75 | 63200 | 3776/4000 | 0 | Per-minute Jobs |
| 21 | Mwe | 309D71E | 0 | 1 | 0 | 3824/4000 | 0 | SYNCCD2430 Helpe |

--More--

Show Stack

A stack is basically a portion of the memory that is used to monitor the internal operations of a program. These stacks are "Last In, First Out" (LIFO) data structures. The **show stacks** command looks at the manner in which the Cisco router's processes and interrupts utilize these stacks. If there was a reboot caused by a crash, then using **show stacks** may reveal the reason for that reboot.

Router> **show stacks**

Minimum process stacks:

| Free/Size | Name |
|-----------|-------------------|
| 2704/4000 | Setup |
| 3256/4000 | Autoinstall |
| 2776/4000 | DNS Snoop |
| 2680/4000 | Init |
| 1720/2000 | LAPB Timer |
| 5400/6000 | BootP Resolver |
| 3460/4000 | RADIUS INITCONFIG |
| 4632/5000 | DHCP Client |
| 3524/4000 | Exec |

Interrupt level stacks:

| Level | Called | Unused/Size | Name |
|-------|--------|-------------|---|
| 1 | 0 | 3000/3000 | CL-CD2430 transmit interrupts |
| 2 | 0 | 3000/3000 | CL-CD2430 receive interrupts |
| 3 | 33 | 2772/3000 | Serial interface state change interrupt |
| 4 | 23 | 2872/3000 | Network interfaces |
| 5 | 10771 | 2896/3000 | Console Uart |

Show Buffers

A buffer is a portion of memory in which data can rest while it waits to catch the next bus out. Buffers are sort of like bus stops, but some are bigger (like a bus station); and some of them are very large, like an airport! The **show buffers** command lets you see the size of the small, middle, big, very big, large, and huge buffers. It also gives statistics on their usage, kind of like baseball scores.

```
Router> show buffers
```

```
Buffer elements:
```

```
500 in free list (500 max allowed)
128 hits, 0 misses, 0 created
```

```
Public buffer pools: Small buffers, 104 bytes
  (total 56, permanent 50):
54 in free list (20 min, 150 max allowed)
87 hits, 2 misses, 0 trims, 6 created
0 failures (0 no memory)
```

```
Middle buffers, 600 bytes (total 28, permanent 25):
28 in free list (10 min, 150 max allowed)
76 hits, 1 misses, 0 trims, 3 created
0 failures (0 no memory)
```

```
Big buffers, 1524 bytes (total 50, permanent 50):
47 in free list (5 min, 150 max allowed)
19 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)
```

```
VeryBig buffers, 4520 bytes (total 10, permanent 10):
10 in free list (0 min, 100 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)
```

```
Large buffers, 5024 bytes (total 0, permanent 0):
0 in free list (0 min, 10 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)
```

```
Huge buffers, 18024 bytes (total 0, permanent 0):
0 in free list (0 min, 4 max allowed)
0 hits, 0 misses, 0 trims, 0 created
0 failures (0 no memory)
```

```
Interface buffer pools: Ethernet0 buffers, 1524 bytes
  (total 32, permanent 32):
8 in free list (0 min, 32 max allowed)
24 hits, 0 fallbacks
8 max cache size, 8 in cache
```

```
BRI0 buffers, 1524 bytes (total 4, permanent 4):
3 in free list (0 min, 4 max allowed)
3 hits, 0 fallbacks
1 max cache size, 1 in cache
```

```
BRI0:1 buffers, 1524 bytes (total 16, permanent 16):
12 in free list (0 min, 16 max allowed)
12 hits, 0 fallback
4 max cache size, 4 in cache
```

```
BRI0:2 buffers, 1524 bytes (total 16, permanent 16):
12 in free list (0 min, 16 max allowed)
12 hits, 0 fallbacks
4 max cache size, 4 in cache
```

```
Serial0 buffers, 1524 bytes (total 32, permanent 32):
7 in free list (0 min, 32 max allowed)
25 hits, 0 fallbacks
8 max cache size, 8 in cache
```

```
Serial1 buffers, 1524 bytes (total 32, permanent 32):
```

```
7 in free list (0 min, 32 max allowed)
25 hits, 0 fallbacks
8 max cache size, 8 in cache
```

```
Serial2 buffers, 1524 bytes (total 8, permanent 8):
6 in free list (0 min, 8 max allowed)
6 hits, 0 fallbacks
0 max cache size, 0 in cache
```

```
Serial3 buffers, 1524 bytes (total 8, permanent 8):
6 in free list (0 min, 8 max allowed)
6 hits, 0 fallbacks
0 max cache size, 0 in cache
```

```
CD2430 I/O buffers, 1524 bytes (total 20, permanent 20):
10 in free list (0 min, 20 max allowed)
10 hits, 0 fallbacks
```

Show Processes CPU

Much like the memory output, the **show processes cpu** command contains a lot of good information but the most important part is the first line. This line shows some the running utilization of the router's CPU. This router has a 5- minute average utilization of 10%, which is not too bad. If you see a router with a 5-minute utilization of over 60%, you might want to do some serious investigating. If you see a router with a 5-minute utilization of over 95%, you may not be able to get processor time to do any investigating.

```
Router# sh proc cpu
```

```
CPU utilization for five seconds: 14%/11%; one minute: 10%;
five minutes: 10%
```

| PID | Runtime (ms) | Invoked | uSecs | 5Sec | 1Min | 5Min | TTY | Process |
|-----|--------------|---------|-------|-------|-------|-------|-----|------------------|
| 1 | 92 | 2297 | 40 | 0.00% | 0.00% | 0.00% | 0 | Load Meter |
| 2 | 1128 | 163 | 6920 | 1.39% | 0.31% | 0.27% | 0 | Exec |
| 3 | 17676 | 384 | 46031 | 1.80% | 0.22% | 0.14% | 0 | Check heaps |
| 4 | 0 | 1 | 0 | 0.00% | 0.00% | 0.00% | 0 | Pool Manager |
| 5 | 4 | 2 | 2000 | 0.00% | 0.00% | 0.00% | 0 | Timers |
| 6 | 4 | 194 | 20 | 0.00% | 0.00% | 0.00% | 0 | ARP Input |
| 7 | 0 | 1 | 0 | 0.00% | 0.00% | 0.00% | 0 | SERIAL A'detect |
| 8 | 0 | 192 | 0 | 0.00% | 0.00% | 0.00% | 0 | IP Input |
| 9 | 20 | 1152 | 17 | 0.00% | 0.00% | 0.00% | 0 | CDP Protocol |
| 10 | 4 | 22 | 181 | 0.00% | 0.00% | 0.00% | 0 | MOP Protocols |
| 11 | 256 | 11693 | 21 | 0.00% | 0.00% | 0.00% | 0 | IP Background |
| 12 | 40 | 2301 | 17 | 0.00% | 0.00% | 0.00% | 0 | TCP Timer |
| 13 | 0 | 1 | 0 | 0.00% | 0.00% | 0.00% | 0 | TCP Protocols |
| 14 | 4 | 1 | 4000 | 0.00% | 0.00% | 0.00% | 0 | Probe Input |
| 15 | 0 | 1 | 0 | 0.00% | 0.00% | 0.00% | 0 | RARP Input |
| 16 | 4 | 1 | 4000 | 0.00% | 0.00% | 0.00% | 0 | BOOTP Server |
| 17 | 20 | 192 | 104 | 0.00% | 0.00% | 0.00% | 0 | IP Cache Ager |
| 18 | 0 | 192 | 0 | 0.00% | 0.00% | 0.00% | 0 | NBF Input |
| 19 | 0 | 2 | 0 | 0.00% | 0.00% | 0.00% | 0 | SPX Input |
| 20 | 0 | 2 | 0 | 0.00% | 0.00% | 0.00% | 0 | DDR Timers |
| 21 | 0 | 1 | 0 | 0.00% | 0.00% | 0.00% | 0 | SNMPConfCopyProc |

--More--

Show CDP Neighbors

The **show cdp neighbors** command shows you all the Cisco equipment to which your router has a direct physical connection. It is used once you have connected your router to other Cisco routers on your network. It uses the Cisco Discovery Protocol, which is proprietary to Cisco and is why it only finds Cisco devices. The command is very useful for troubleshooting networks. If **show cdp neighbors** doesn't show a connection, basically you aren't connected (to a Cisco device).

CDP is a layer 2 protocol. It is only used for informational updates on directly connected links. What this means to us is that when we look at CDP information on a Cisco router, it will tell us a lot of information about the other Cisco equipment connected directly to the same networks as the router at which we are looking. Since this is a layer 2 protocol, all communications for CDP are broadcast-based, and these broadcasts are sent out every 60 seconds by default. Let's take a look at what we can see from our test network environment.

The **sh cdp ?** command shows us what command arguments are available:

```
Router# sh cdp ?
  entry      Information for specific neighbor entry
  interface  CDP interface status and configuration
  neighbors  CDP neighbor entries
  traffic    CDP statistics
<cr>
```

The real value of the **sh cdp** commands is being able to see the status of the network, the devices to which you are connected, and what their capabilities are. Each CDP status will show us three basic things: accessibility, capabilities, and device type. A simple example on the router from our example shows the following:

```
Router# sh cdp nei
```

```

Capability Codes: R - Router, T - Trans Bridge,
                  B - Source Route Bridge,
                  S - Switch, H - Host, I - IGMP,
                  r - Repeater
Device ID  Local  Holdtme Capability Platform Port ID
          Inrfce
Router2    Ser 0    144   R           2500     Ser 1
066538247  Eth 0    159   T B S       WS-C5505 4/8
Router#

```

What this shows us is basic connection information about to what a router is locally connected and with what it is communicating. At the moment of this capture, Router1 is connected to a 2500 router named Router2, as well as being connected to a WS-C5505, which is a Catalyst 5505 switch. As you can see, you can get connectivity information on all Cisco products that have CDP enabled. As of IOS version 10.3 and later, CDP is enabled by default on all interfaces.

To look at an individual entry in the CDP neighbor table, use the command where name is the router name or other Cisco device found in the neighbor table. The name is case sensitive. This command is useful if the neighbor table is large and you want to get detailed information about a single neighbor without having to view the details of all neighbors.

```
Router# show cdp entry Router1
```

```

Device ID: Router1
Entry address(es):
  IP address: 192.168.2.2
Platform: cisco 2500, Capabilities: Router
Interface: Serial0, Port ID (outgoing port): Serial1
Holdtime : 147 sec

```

```

Version :
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L),
Version 11.2(18), RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.
Compiled Mon 05-Apr-99 20:23 by jaturner

```

The **show cdp neighbors detail** command gives the same output as **show cdp entry**, except it shows detail for the connected routers and other devices.

```
Router# sh cdp nei det
```

```

Device ID: Router1
Entry address(es):
  IP address: 192.168.2.2
Platform: cisco 2500, Capabilities: Router
Interface: Serial0, Port ID (outgoing port): Serial1
Holdtime : 147 sec

```

```

Version :
Cisco Internetwork Operating System Software
IOS (tm) 2500 Software (C2500-JS-L),
Version 11.2(18), RELEASE SOFTWARE (fc1)
Copyright (c) 1986-1999 by cisco Systems, Inc.

```

```
Compiled Mon 05-Apr-99 20:23 by jaturner
```

```

Device ID: 066538247(GCD5505)
Entry address(es):
  IP address: 10.200.1.5
Platform: WS-C5505,
  Capabilities: Trans-Bridge Source-Route-Bridge Switch
Interface: Ethernet0, Port ID (outgoing port): 4/8
Holdtime : 161 sec

```

```

Version :
WS-C5505 Software, Version McpSW: 5.1(1) NmpSW: 5.1(1)
Copyright (c) 1995-1999 by Cisco Systems

```

Showing the CDP neighbor detail statistics is a very handy troubleshooting tool if you know you are having problems with a certain IOS software version. You can use this command on core routers to check the software versions of each directly connected router.

If you want to see what the timer settings are for CDP broadcast updates, you can enter the following:

```

Router# sh cdp int
Ethernet0 is up, line protocol is up
  Encapsulation ARPA
  Sending CDP packets every 60 seconds

```

```

Holdtime is 180 seconds
Serial0 is up, line protocol is up
  Encapsulation HDLC
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds
Serial1 is administratively down, line protocol is down
  Encapsulation HDLC
  Sending CDP packets every 60 seconds
  Holdtime is 180 seconds

```

The CDP timers (update interval and hold time) are global parameters. When they are changed, all interfaces running CDP are changed. The command to change the periodic update interval uses seconds. Without using the command the default value is 60. The command to change the hold time also uses seconds. The default value is 180. Remember that when timers are changed on one device, they should be changed on all the rest of the devices to help prevent neighbor-table synchronization problems.

```

Router# cdp timer seconds
Router# cdp holdtime seconds

```

CDP can be enabled and disabled either on individual interfaces or globally. If you want to turn off CDP on an individual interface, use the interface configuration mode command **no cdp enable**. To turn it back on, issue the **cdp enable** command. To turn off CDP on all interfaces simultaneously, issue the global configuration command **no cdp run**. Issue the **cdp run** command to turn CDP back on.

Debug Commands

Just as the show commands are useful for verifying router status and diagnosing configuration problems, the various debug commands can be helpful too. The **debug** command allows us to see what the IOS is doing as things happen, and it is normally used for troubleshooting and experimenting. We can turn on many different types of debug activities. Each one shows us something different about what is going on inside a router. To see the possible variations of the debug command, use the inline, context-sensitive help. Start by typing **debug ?**, and extend the command from there.

Debug output, by default, is logged to the console line and to terminal lines that have the monitor capability turned on. When we want to view debug output in a Telnet session, we can give our VTY the monitor capability by issuing the command **terminal monitor** in privileged mode.

There are a lot of debug commands that you can safely use in a classroom or test lab to show you what a router actually does while routing. Here's a tip: before doing any debug commands, type in the **no debug all** command. That way if something does go wrong while debugging and your router starts spewing out so much information that you can not type anything into the command line, all you need to do is to hit the UP arrow and press Enter. That'll turn the debug off.

To see all the debug commands just type **debug**, a space, and a question mark.

```

Router# debug ?

aaa                AAA Authentication,
                   Authorization and Accounting
access-expression  Boolean access expression
all                Enable all debugging
alps               ALPS debug information
apollo             Apollo information
apple              Appletalk information
arap               Appletalk Remote Access
arp                IP ARP and HP Probe transactions
aspp               ASPP information
async              Async interface information
backup             Backup events
bri-interface      bri network interface events
bsc                BSC information
bstun              BSTUN information
callback           Callback activity
cdp                CDP information
chat               Chat scripts activity
clns               CLNS information
cls                CLS Information
cns                CNS Debugging
compress           COMPRESS traffic
condition          Condition
confmodem          Modem configuration database
cpp                Cpp information
custom-queue       Custom output queueing
decnet             DECnet information
dhcp               DHCP client activity
dialer             Dial on Demand
dls                Data Link Switching (DLSw) events
dnsix              Dnsix information
domain             Domain Name System
drip               DRIP debug information

```

| | |
|--------------------|--|
| dspu | DSPU Information |
| dxi | atm-dxi information |
| eigrp | EIGRP Protocol information |
| entry | Incoming queue entries |
| ethernet-interface | Ethernet network interface events |
| frame-relay | Frame Relay |
| fras | FRAS Debug |
| fras-host | FRAS Host Debug |
| funi | FUNI interface packets |
| gssapi | GSSAPI debugs |
| interface | interface |
| ip | IP information |
| ipx | Novell/IPX information |
| isdn | ISDN information |
| isis | IS-IS Information |
| kerberos | KERBEROS authentication and authorization |
| lapb | LAPB protocol transactions |
| lat | LAT Information |
| ldap | LDAP debug commands |
| lex | LAN Extender protocol |
| list | Set interface or/and access list for the next debug command |
| llc2 | LLC2 type II Information |
| lnm | Lan Network Manager information |
| lnx | generic qlc/llc2 conversion activity |
| local-ack | Local ACKnowledgement information |
| management | Management applications debugging |
| modem | Modem control/process activation |
| mop | DECnet MOP server events |
| nbf | NetBIOS information |
| ncia | Native Client Interface Architecture (NCIA) events |
| netbios-name-cache | NetBIOS name cache tracing |
| nhrp | NHRP protocol |
| ntp | NTP information |
| nvramp | Debug NVRAM behavior |
| packet | Log unknown packets |
| pad | X25 PAD protocol |
| pcbus | PCbus interface information |
| ppp | PPP (Point to Point Protocol) information |
| printer | LPD printer protocol |
| priority | Priority output queueing |
| probe | HP Probe Proxy Requests |
| qlc | qlc debug information |
| radius | RADIUS protocol |
| rif | RIF cache transactions |
| rtr | RTR Monitor Information |
| sdlc | SDLC information |
| sdllc | SDLLC media translation |
| serial | Serial interface information |
| sgbp | SGBP debugging |
| smf | Software MAC filter |
| smrp | SMRP information |
| sna | SNA Information |
| snapshot | Snapshot activity |
| snmp | SNMP information |
| source | Source bridging information |
| spantree | Spanning tree information |
| sscop | SSCOP |
| standby | Hot standby protocol |
| stun | STUN information |
| tacacs | TACACS authentication and authorization |
| tarp | TARP information |
| tbridge | Transparent Bridging |
| Telnet | Incoming Telnet connections |
| tftp | TFTP debugging |
| token | Token Ring information |
| translate | Protocol translation events |
| tunnel | Generic Tunnel Interface |
| v120 | V120 information |
| vg-anylan | VG-AnyLAN interface information |
| vines | VINES information |
| vpdn | VPDN information |
| vprofile | Virtual Profile information |
| vtemplate | Virtual Template information |
| x25 | X.25, CMNS and XOT information |
| x28 | X28 mode |
| xns | XNS information |

As you can see, there are lots of debug commands to choose from. The command **debug all** is too verbose -- it makes the information you're looking for hard to find in all the detailed data about literally everything in the router. We need to narrow down our focus a bit. Let's say we've been having a problem with our Ethernet interface. If we do a **debug ethernet-interface** command we can see what is going on.

```
Router# debug ethernet-interface
Ethernet network interface debugging is on
Router#
00:29:07: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
00:29:17: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
00:29:27: %LANCE-5-LOSTCARR: Unit 0,
  lost carrier. Transceiver problem?
```

As you can see, the Ethernet interface is definitely not working. Discovering what is not working is the whole key to troubleshooting. In this case, we know it's because the Ethernet is not connected to a network segment.

Fallback

You can define a fallback sequence of "boot system" commands that specify alternate ways for your router to boot if your router can't find the IOS on the first try. The boot system commands are placed in the startup configuration file and will only be used if the router is configured for normal boot sequence.

For instance, the boot system commands can specify that the router will load first from Flash memory, next from a TFTP server, and finally from ROM. In this case, the TFTP server is a fallback in case the Flash memory is corrupted and ROM is a fallback in case the TFTP server is unavailable.

After the Bootstrap Startup Program during the boot process (see [Router Boot Sequence](#), above), the router scans the startup configuration in NVRAM for boot system commands to find out whether it should get its IOS image from a location other than the first file in Flash memory. If it finds boot system commands, it executes them in sequence until it finds a valid image that it can load. If there are no boot system commands, the router will attempt to load the first IOS image it finds in Flash.

Finally, if that fails, it will use the image in ROM as a fallback. This minimally featured image will allow IP addresses to be assigned to interfaces and will allow you to use ping and TFTP, but there's not much else in there. The prompt is

```
Router(boot)>
```

The ROM monitor is an unfriendly place to be, the commands are arcane, and the prompts give you no help.

We'll discuss how to write boot system commands in the next section.

The Configuration Register

During the boot process we can have the router look into the startup configuration for where to find the IOS image or we can have the router bypass this step and have it look in a specific place. The primary method for dictating how a router behaves on startup is the configuration register, also known as the config register. The config register on a Cisco router is a 16-bit register that tells the router what to do when booting.

With the config register, you can force the router to boot in ROM monitor mode, select boot source and filename, enable and disable the break function during boot, control broadcast address mapping, and control the load source of IOS. It is for this last reason that we will examine the last 4 bits of the configuration register. These last 4 bits are called the "boot field." Since the configuration register is shown in hexadecimal numbers, the last 4 bits will be represented by only one digit.

Another reason the config register is important is that when we forget our password, the config register plays a major part in password recovery. This is a topic for another Tutorial, but in the meantime the Cisco web site has [more information about password recovery](#). (is not associated with Cisco.)

If the boot field is set to hex 00, then the router will boot to the ROM monitor mode on reload or power up. If this field is set to hex 01, the system will boot fully and load the first IOS image in Flash memory. A hex value of 02 is the most flexible, it allows default image booting from Flash as well as enabling "boot system" commands in the startup-config file. Table 3 shows the values and functions of the boot field.

Table 3. Boot Field Positions and Functions

| Boot field value | Function |
|------------------|---|
| 00 | Boot to ROM monitor mode |
| 01 | Boot with default software contained in ROM |
| 02-0F | Enables default booting from Flash, then enables "boot system" commands, then implicitly describes a default netboot filename |

When the boot field is between 02 and 0F, the router boots from Flash. If it doesn't find an IOS image in Flash it will look to the boot system commands in the startup configuration. If it doesn't find one there, it can look outside of the router to the network. In this last case, the router will need a network filename or netboot filename. The default netboot filename is derived by taking the boot field value and pairing that up with the processor name (name=cisco<n>-processor_name where n is some number between 2 and 15).

If we put all these possibilities together, the config register will become 0x2102. Then the IOS will boot from Flash if a valid file exists or will enable "boot system" commands in the config file; if there is no valid software image in Flash, it will attempt to netboot the file name cisco2-2500.

By the way, to display the content of the configuration register, use the **show version** command. (See [Show Version](#), above.)

Changing the Configuration Register

In order to modify the config register on your router, you **must** be connected to the console port. This cannot be performed via a telnet session to the router.

There are a couple of methods for changing the config register on a Cisco router. One of those methods can be done through regular config mode on the router. In order to change the config register, you must login to the router, enter privileged mode, and then enter config mode. Once there, you will simply enter a single command, **config-register**, that changes the config register (by the way, the shorthand for this is **conf**). In this command, the expected input is expressed in hex.

Since there is no difference between decimal and hex until digits get above 9, the way to denote hexadecimal notation is with the prefix 0x. For example, the notation for the configuration that would boot the router normally but ignore NVRAM is 0x2142, as shown below:

```
Router> en
Router# config t
Router(config)# config-register 0x2142
Router(config)# ctrl-z
Router#
```

This command sequence will change the router's config register from whatever it is to 0x2142. Now let's check to verify; type **sh ver**, Enter, and then hit the [space] bar. This will show the software version as well as the config register. At the bottom of the second screen of output, you will see the line

```
Configuration register is 0x2102
(will be 0x2142 at next reload).
```

If you make a change to the config register, you will see the latter parenthesized statement. If you enter the config register command in config mode, but do not actually change the register, you will not see this output. The config register we have entered will cause the router to boot without reading NVRAM for the startup config, and we don't really want that, so let's change it back:

```
Router> en
Router# config t
Router(config)# config-register 0x2102
Router(config)# ctrl-z
Router#
```

Now when we perform a **sh ver**, we will see that the configuration register is 0x2102. There will no longer be the added statement telling us what the config register will be on the next reload.

NOTE: For heavens sake, if you do change the configuration register for any reason, be sure to set it back to what it was before so the router will use its proper configuration and find its IOS image properly the next time it boots up.

Booting from ROM (Boot Field = 01)

If the boot field of the config register is set to 01 (example: 0x2101) the router will boot from a default IOS image that is stored in ROM. In most cases, the software that is loaded in ROM is an earlier version and only has limited functionality. The reason for having this capability is in the case of a problem with the IOS software that resides in Flash.

If you have a problem with that image, you can boot to the default image and TFTP a new image to the router. If for some reason you are working on a router and happen to see a (boot) prompt (example: Router(boot)>), this means that the router is running in ROM monitor mode. This will give you a very quick check to see what the problem is with the router. Check the router and see what the config register value is. If it is 0x2101, then change the config register and reboot. If the config register is 0x2102, you know that something has happened to the software image loading from Flash.

Booting from Flash (Boot Field = 02)

If the config register has the boot field set to a value of 02, the router will boot from Flash. It is possible to have multiple images in Flash, as well as multiple partitions in Flash. If the boot field is set to 02, the router will look at the startup config to see if there is a boot system command telling it what file to load from Flash. If there are no boot system commands, the router will use the first valid system image in Flash. If there are no valid system images in Flash, then the router will attempt a netboot.

With a value of 02 in the boot field, a Cisco 2500 will send out broadcasts on the directly connected networks to retrieve an IP address and look for a

TFTP server with the file cisco2-2500 on it. If none of the above circumstances are met, then the bootstrap program will check the 13th bit of the config register. If the 13th bit has an "on" value, the router will load the default software out of ROM.

If there are multiple valid system images in Flash, boot system commands can be entered into the startup configuration to tell the router which one to use. The following is the syntax for configuring the router to boot from Flash and look for c2500-d-l.120-5.bin first, c2500-d-l.112-19.bin second, then boot from ROM:

```
Router# conf t
Router(config)# boot sys Flash c2500-d-l.120-5.bin
Router(config)# boot sys Flash c2500-d-l.112-19.bin
Router(config)# boot sys rom
Router(config)# [ctrl-z]
Router# copy run start
```

Boot system commands will be executed on startup in the order in which they are entered. In the above example, the system will try to boot with the 12.0 release software first, then the 11.2 software, and then revert to ROM.

There is another boot system command that tells the router to boot from the network (from a TFTP server). The syntax is similar, and the following will load the same images from a TFTP server with the IP address of 192.168.1.100.

```
Router# conf t
Router(config)# boot sys tftp c2500-d-l.120-5.bin
                  192.168.1.100
Router(config)# boot sys tftp c2500-d-l.112-19.bin
                  192.168.1.100
Router(config)# boot sys rom
Router(config)# [ctrl-z]
Router# copy run start
```

In this example, we explicitly told the router to look for the TFTP server at IP address 192.168.1.100. If the TFTP server was located on a LAN local to the router, we could have left this information off, and the router would have sent out a broadcast and located the server on its own. Another note regarding this type of configuration is that since there are options to booting, the router has a fallback. If the first option doesn't work, it can fall back on the second option, then the third, which is boot with the default software image in ROM. (See [Fallback](#), above).

Appendix A. Review Questions and Answers

Review Questions

(Answers are provided at the end.)

Answer the following questions as either True or False.

1. The most common method for a router to find the configuration commands are the ones saved in NVRAM.
2. You can specify enabled config-mode boot system commands to enter fallback sources for the router to use in sequence.
3. In order to check any config-register setting you use either the **show running-config** or **show startup-config** commands.
4. The system image in ROM contains the full set of Cisco IOS software, which is equal to the one found in Flash.
5. The system image stored in Flash memory can be copied to ROM with the command **copy flash rom**.
6. By default, Cisco routers are DTE devices, but sometimes we need to turn them into DCE devices.
7. The configure console command is used to configure manually from the console terminal.
8. You can configure a message-of-the-day banner to be displayed on all connected terminals using the **banner config** command.
9. The first global parameter for which you are prompted by a router during Setup allows you to set the router's host name.
10. If Flash memory is corrupted, or its IOS is missing and the network server fails to load an IOS image, booting from ROM is the final bootstrap option in software.
11. During router Setup you are not prompted for parameters for each installed interface, you must enter these separately using the **configure interfaces** command.
12. During router setup you need not enter an enable secret password, but you must enter just the enable password.

13. Flash memory holds the operating system image and microcode, allowing updates to software without removing and replacing chips on the processor.
14. CDP runs over a data link layer, connecting lower physical media and upper-network-layer protocols.
15. Default values for CDP timers set the frequency between CDP updates and for aging CDP entries.
16. The **show version** command displays information about the Cisco IOS software version that is currently running on the router.
17. CDP can only discover information about directly connected Cisco devices if they are using the same protocol suite.
18. The TFTP server can be another router, or it can be a host computer system.
19. If no free Flash memory space is available, or if the Flash memory has never been written to, the erase routine is usually required before new files can be copied to it.
20. A router can only have one incoming Telnet session at a time.
21. The terminal editing command enables advanced editing.
22. Commands available in privileged mode are a subset of the commands available in the user mode.
23. From the user mode, you can also access global configuration mode and the other specific configuration modes.
24. You can press Control-C to terminate the setup process and start over at any time.
25. For many of the prompts in the system configuration dialog of the setup command facility, default answers appear in square brackets ([]) following the question.

Review Answers

- | | |
|-----------|-----------|
| 1. True | 14. True |
| 2. False | 15. True |
| 3. False | 16. True |
| 4. False | 17. False |
| 5. False | 18. True |
| 6. True | 19. True |
| 7. False | 20. False |
| 8. False | 21. True |
| 9. True | 22. False |
| 10. True | 23. False |
| 11. False | 24. True |
| 12. False | 25. True |
| 13. True | |

Appendix B. Cisco Router Series

It can be useful to think of Cisco routers as having been introduced in several product generations. This isn't completely pure, because some devices called "switches" actually route. In general, however, it's useful to think of:

- First generation routers: totally obsolete proof-of-concept
- Second-generation: IGS, CGS, MGS, and AGS (which ran a recognizable IOS)
- Third-generation: 2500, 4000, and 7000, running at least IOS 9.0
- Fourth-generation: broadband access routers for ISDN, xDSL, CATV, etc., (all running IOS), 1600, 2600, and advanced 7100/7200, 7500. "Layer 3 switching" in 5000/5500 switches.
- Fifth-generation: very high capacity including 10000 and 12000; Layer 3 switching in 6000/6500/8500 switches.

Having looked at the most common Cisco router, we can now discuss the different models of Cisco routers and give some particulars about each one. There are several varieties of Cisco routers. The relevant router models are the 2500, 4000, 7000, and 7500 series. The 4000 is the next step up after the 2500 series in Cisco's product line. The following lists show some of the Cisco routers and give their primary uses. When the description of any Cisco router series includes the term "slot" you should think of an opening where a removable card or component can be inserted.

Series 700

This family of products is ISDN dial-on-demand routers for home offices and telecommuters (base \$400 to \$800).

- 761 - One Ethernet port plus ISDN BRI S/T
- 762 - One Ethernet port plus ISDN BRI NT1

- 765 - One Ethernet port plus ISDN BRI S/T plus two POTS
- 766 - One Ethernet port plus ISDN BRI NT1 plus two POTS
- 771 - Four Ethernet ports plus ISDN BRI S/T
- 772 - Four Ethernet ports plus ISDN BRI NT1
- 775 - Four Ethernet ports plus ISDN BRI S/T plus two POTS
- 776 - Four Ethernet ports plus ISDN BRI NT1 plus two POTS

Series 1600

This family of routers is a slightly scaled down version of the more expensive and popular 2500 Series family. Generally, it has one serial and one Ethernet port rather than two like the 2500 has. Unlike most of the 2500 family, the 1600's have a WAN module slot for flexibility.

- 1601 - One Ethernet port and one serial port
- 1602 - One Ethernet port and one 56K CSU/DSU
- 1603 - One Ethernet port and one ISDN BRI S/T
- 1604 - One Ethernet port and one ISDN BRI NT1
- 1605R - Two Ethernet ports

Series 2500

The 2500 Series is the world's most popular small-to mid-sized router. There are so many types of routers available in this series that it is helpful to organize the list into several groups as is done below:

Single LAN

- 2501 - One Ethernet port, two serial ports
- 2502 - One Token Ring port, two serial ports
- 2503 - One Ethernet port, two serial ports, one ISDN BRI
- 2504 - One Token Ring port, two serial ports, one ISDN BRI
- 2520 - One Ethernet port, two serial ports, one ISDN BRI, 2 low speed serial
- 2521 - One Token Ring port, two serial ports, one ISDN BRI, 2 low speed serial
- 2522 - One Ethernet port, two serial ports, one ISDN BRI, 8 low speed serial
- 2523 - One Token Ring port, two serial ports, one ISDN BRI, 8 low speed serial

Dual LAN

- 2513 - One Ethernet, one Token Ring, two serial ports
- 2514 - Two Ethernet ports, two serial ports
- 2515 - Two Token Ring ports, two serial ports

Router/Hub Combo

- 2505 - Eight-port Ethernet hub, two serial ports
- 2507 - 16 port Ethernet hub, two serial ports
- 2516 - 14 port Ethernet hub, one ISDN BRI, two serial ports

Access Servers

- 2509 - One Ethernet, eight asynch, two serial ports
- 2511 - One Ethernet, 16 asynch, two serial ports
- 2512 - One Token Ring, 16 asynch, two serial ports

Series 2600

This family of routers is very comparable to the popular 2500 series, except they have slots, including one Module slot for features like voice and fax, and two WAN slots for options like built-in CSU/DSU, ISDN, serial ports, asynch ports, and so forth.

- 2610 - one Ethernet, one Module slot, two WAN slots
- 2611 - two Ethernet, one Module slot, two WAN slots
- 2612 - one Ethernet, one Token Ring, one Module slot, two WAN slots
- 2613 - one Token Ring, one Module slot, two WAN slots
- 2620 - one Ethernet 10/100, one Module slot, two WAN slots
- 2621 - two Ethernet 10/100, one Module slot, two WAN slots

Series 3600

The 3600 Series is a multifunction platform that combines dial access, routing, LAN-to-LAN services, and multifunction integration of voice, video, and data in the same device. It is somewhat similar to the 2600 Series in regard to available slot options except it has a much richer list of those options. Also, the 3600 is a router meant to be installed in a larger network, like a WAN headend where the core of the WAN is located.

Series 4000

The third-generation Cisco 4000 series consists of the Cisco 4000-M, the Cisco 4500-M, and the Cisco 4700-M. The Cisco 4000 series is used in middle-size networks and at the distribution layer of large internetworks. All models provide a configurable modular router platform using network processor modules including FDDI, ISDN BRI, ISDN PRI, Ethernet 100baseT, Token Ring, HSSI, and ATM. The Cisco 4000 series routers support up to three network processor modules at a time. The Cisco 4700-M contains a 133-MHz RISC microprocessor, 16 to 64 MB main memory, and a 512-KB secondary cache. The faster speed of the Cisco 4700-M allows higher throughput for high-speed interfaces. The 512-KB secondary cache is useful for process switching applications such as compression and encryption.

In new installations, the 4000 series has been superceded by the 3600 series.

Be sure to pay attention to the model number. The routers are the 4000, 4500, and 4700. There is a new series of gigabit layer 2 switches called the 4000 series.

AGS+ and Series 7000

Some Cisco routers were the most powerful of their time, but have been superceded by newer models. The AGS+ is a "second generation" router, which you may find available cheaply. It generates substantial heat and noise, and will run IOS versions no later than early 11. Nevertheless, many certification candidates find it cost-effective in their labs. Be sure that you buy a version with the CSC/4 processor if you expect to run any recent IOS. The AGS+ was the first Cisco router that could use different processors for path determination and packet forwarding.

The third-generation replacement for the AGS+ was the Cisco 7000 series of multi-protocol routers, including the Cisco 7000 and the Cisco 7010. The 7000 and 7010 differ only in their number of card slots, Network interfaces reside on modular interface processors, which provide a direct connection between the high-speed Cisco Extended Bus (CyBus) and the external network. Distributed processing is accomplished by the Route Processor (RP), Switch Processor (SP), and Silicon Switch Processor (SSP).

Series 7100 and 7200

The fourth-generation Cisco 7200 series of multi-protocol routers delivers the performance, port density, and availability typically associated with high-end systems. This router is used as a high-speed backbone aggregation router for high-speed enterprise interconnectivity. It is used for a WAN edge concentrator at the backbone and IBM interconnectivity. The 7100 has additional hardware to assist it in encryption for virtual private networks.

Series 7500 and 10000

The Cisco 7500 was designed to meet the demands of emerging high-end application environments -- in terms of density, performance, and system availability. Like the 7200 series, this series is used for high-speed backbone aggregation needed for high-speed enterprise interconnectivity. It is used for a WAN edge concentrator at the backbone.

Series 12000

Cisco's new family of gigabit switch routers (GSR) provides high-performance solutions ranging from five to 60 Gbps for Internet and large-scale WAN intranet backbone applications.

Basic Router Operation Labs

Basic Router Operation Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

In this lab you will complete the following tasks:

- Back up a router configuration file to a TFTP server
- Back up a router IOS software image to a TFTP server
- Restore a router configuration file from a TFTP server
- Restore a router IOS software image from a TFTP server

Basic Router Operation Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Backing Up and Restoring Router Configuration Files and IOS Software Images

[Objectives](#)

[Setup](#)

[TIP](#)

[Scenario](#)

[Task 1: Back up a router configuration file to a TFTP server](#)

[Step 1-1](#)

[Step 1-2](#)

[Step 1-3](#)

[Step 1-4](#)

[Task 2: Back up a router IOS software image to a TFTP server](#)

[Step 2-1](#)

[Step 2-2](#)

[Step 2-3](#)

[Task 3: Restore a router configuration file from a TFTP server](#)

[Step 3-1](#)

[Task 4: Restore router IOS software image from a TFTP server](#)

[Step 4-1](#)

[Solutions](#)

[Task 1, Step 2.](#)

[Task 1, Step 3.](#)

[Task 1, Step 4.](#)

[Task 2, Step 1.](#)

[Task 2, Step 2.](#)

[Task 3, Step 1.](#)

[Task 4, Step 1.](#)

Complete this lab to practice what you learned in the Basic Router Operation Tutorial.

Objectives

In this lab you will complete the following tasks:

- Back up a router configuration file to a TFTP server
- Back up a router IOS software image to a TFTP server
- Restore a router configuration file from a TFTP server
- Restore a router IOS software image from a TFTP server

Setup

Your router should have at least a basic configuration from either completing the router's setup script or manually configuring your router similar to the sample configuration in the Basic Router Operation Tutorial.

You should also install and configure a TFTP server in your lab network. This is fairly easy to do, and any PC or laptop can function as your TFTP server for the purposes of this lab. For more information about finding TFTP software, see Backing Up Router Configuration Files in the Basic Router Maintenance and Troubleshooting section of the Basic Router Operation Tutorial.

Scenario

You've configured your router, and now you want to save your configuration somewhere other than NVRAM. You also want to back up your router's IOS software image in case your router suffers a critical failure. Finally, you want to test these backups by restoring them to your router.

TIP

Many TFTP server implementations require you to access files, for both upload and download, by their fully qualified name -- not their name relative to a directory.

Task 1: Back up a router configuration file to a TFTP server

Step 1-1

Enter privileged EXEC mode.

Step 1-2

Save your running configuration to NVRAM.

What command saves your running configuration to NVRAM?

Step 1-3

Backup your running configuration to the TFTP server.

What command will back up your running configuration to the TFTP server?

Refer to the Backing Up Router Configuration Files section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Give your configuration file a unique name.

Step 1-4

Find and view your configuration file on the TFTP server.

What is different about the configuration file on the TFTP compared to your original running configuration? (Do a **show run** on your router if necessary.)

Task 2: Back up a router IOS software image to a TFTP server

Step 2-1

Look at the contents of Flash memory.

What command shows you the contents of Flash memory?

How many files are currently in Flash memory?

List the files currently in Flash memory:

What file, ending in .bin, will always be found in the Flash memory of newly configured routers?

How many bytes of Flash memory is the .bin file taking up?

How many bytes of Flash memory are left?

Step 2-2

Back up your IOS software image to your TFTP server.

What command backs up your IOS software image to your TFTP server?

Refer to the Backing Up Software Images section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name of the .bin file you wrote above as the source file name.

Step 2-3

Find your IOS software image on the TFTP server to verify that it transferred correctly.

Task 3: Restore a router configuration file from a TFTP server

Step 3-1

Restore the configuration you saved to the TFTP server in Task 1 to your router's current running configuration.

What command restores the configuration you saved to the TFTP server in Task 1 to your router's current running configuration?

Refer to the Backing Up Router Configuration Files section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name you gave your configuration file.

What command would you enter to restore the configuration from the TFTP server directly to your router's startup configuration?

Task 4: Restore router IOS software image from a TFTP server

Step 4-1

Restore the IOS software image you saved to the TFTP server in Task 2 to your router's Flash memory.

What command restores the IOS software image you saved to the TFTP server in Task 2 to your router's Flash memory?

Refer to the Backing Up Software Images section of the Basic Router Operation Tutorial for the correct responses to the router questions. Make sure you enter the IP address you configured for your TFTP server when prompted for the address of the remote host. Use the name of the .bin file you wrote in above as the source file name.

Why would the router need to erase Flash memory in order to restore an IOS software image from a TFTP server?

How does the router indicate that Flash memory is being erased?

How does the router indicate that the IOS is being copied?

Solutions

Task 1, Step 2.

- copy run start

Task 1, Step 3.

- copy run tftp

Task 1, Step 4.

- The configuration file on the TFTP server will have no comments [lines beginning with an exclamation point (!)] in it. The comments get stripped out when being transferred to the TFTP server. You can add them back in on your TFTP server with a regular text editor.

Task 2, Step 1.

- show flash

The number and name of files currently in Flash will vary from router to router. On newly configured routers the only file in Flash is the system image file (the file that ends in .bin). The number of bytes of Flash memory that the system image file takes up will vary from router to router. There should be at least several megabytes of Flash memory still available, however.

Task 2, Step 2.

- copy flash tftp

Task 3, Step 1.

- copy tftp run

- copy tftp start

Task 4, Step 1.

- copy tftp flash

Note the size of the IOS software image from Task 2, Step 1. If there is not enough room in Flash memory to save a second copy of the IOS software image, it will need to erase the Flash to make room.

The letter 'e' is output to the screen to indicate that Flash memory is being erased.

Exclamation points are output to the screen to indicate that the IOS is being copied.

Firewalls Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Firewalls

The Tutorial portion of this Study Guide begins with a very brief overview of what a firewall is and the general functions available in most common firewalls. It then discusses firewall features available in the current (12.x) releases of IOS, gives some general guidelines for "hardening" a router so that it can act as a firewall, and examines the four major IOS enhancements that make up Cisco's Secure Integrated Software (formerly called the "IOS Firewall Feature Set"). The Tutorial concludes with an examination of the features of the PIX, Cisco's dedicated firewall.

Firewalls

Tutorial
Lab Abstract
Lab Scenario

Firewalls Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Building a Firewall: Three Cisco Offerings

[Introduction](#)

[What Happens When You Have Security?](#)

[Firewall Components](#)

[Core Capabilities](#)

[Authentication mechanism](#)

[Packet filtering](#)

[Application gateway](#)

[Other Security Features of Firewalls](#)

[Firewall Platform Considerations](#)

[IOS Security Features](#)

[Guidelines for Configuring a Firewall](#)

[Sample Firewall Router Configuration](#)

[IOS Security Feature -- Dynamic ACLs](#)

[IOS Security Feature -- Network Address Translation](#)

[The IOS Firewall Feature Set](#)

[The PIX Firewall](#)

[Conclusion](#)

[References](#)

Introduction

Security, computer security specifically, seems to be a popular topic these days. Now that Internet access is integral to the operations of many companies, corporate America has begun to see the potential benefits that can result from a shared electronic communications medium. They are also becoming more aware that significant risks -- largely unknown and generally not understood -- result from having a link to the rest of the world.

This makes Internet security difficult. Companies want to be sure that their data is safe but they don't want the inconvenience (in terms of intrusions, processing delays, etc.) that security precautions invariably cause. They are also often loath to spend money on something from which they can see no clear benefit.

The goal of network security is to insure data confidentiality, integrity, and access control. This means that sensitive information is not disclosed to unauthorized agents; that data is not lost, manipulated, or unavailable when needed; and that data is not accessed by anyone without the proper credentials. Clearly these are issues that are a concern for intra- as well as internets, but the focus of this paper is on safeguarding external access.

In general, security threats stem from three sources: policy failures, configuration failures, and failures of the underlying protocols, operating systems, or procedures. As an example, a firewall may be vulnerable to external attacks because

- there was no formal policy, or a formal policy existed but wasn't implemented uniformly
- generic user accounts with default passwords were left open (a configuration failure)
- the firewall's operating system had a vulnerability (a failure of the underlying system).

This paper will start with a very brief overview of what a firewall is and the general functions available in most common firewalls. We will then discuss firewall features available in the current (12.x) releases of IOS and give some general guidelines for "hardening" a router so that it can act as a firewall. As part of this discussion of IOS features we'll discuss and give configuration examples for two IOS functions that are frequently used in firewalls but not in internal routers -- Dynamic ACLs and Network Address Translation (NAT). We'll then look at the four major IOS enhancements that make up Cisco's Secure Integrated Software (formerly called the "IOS Firewall Feature Set") and look at a few examples that show the functions of these features. Finally we'll look at features of the PIX, Cisco's dedicated firewall.

Two other popular topics related to security -- encryption and advanced authentication methods -- are beyond the scope of this paper and may be discussed in future Issues.

What Happens When You Have Security?

(from H. Berkowitz, *WAN Survival Guide*, Wiley (Fall 2000))

I have found it quite useful to group together "faults" and "security incidents," because they both really deal with the same problem: ensuring that legitimate users can use the resources they need. Protecting against denial of service attacks, while usually considered a security measure, is just as much a fault tolerance mechanism as a security mechanism. Fault-tolerant design and network management tools help protect against service failures due to errors and disasters. Additional security services deal with an additional problem: that unauthorized users do not have access to services or data.

- Disclosure of information. *Confidentiality* mechanisms protect against this threat.
- Unauthorized resource use. *Access authentication* and *access control* help here.
- Alteration/replay/duplication of information. *Unitary integrity* protects against changes of single records, while *sequential integrity* protects against insertion or deletion of records in a sequence.

I find it useful to begin my security planning not so much with threats, but in a more positive manner, considering the characteristics of security success. Dennis Branstad created the excellent 5-S mnemonic for potential aspects of a secure communication. Not every application will require every aspect of the checklist:

- *Sealed*: protected against unauthorized modification. *Unitary integrity* mechanisms seal messages.
- *Sequenced* to protect against unauthorized loss or modification. *Sequential integrity* mechanisms ensure that the sequencing of records is not altered.
- *Secret* so it cannot be disclosed without authorization. *Confidentiality* mechanisms protect the information from unauthorized eyes.
- *Signed* to assure the correct identity of its sender. *Authentication* and *digital signature* mechanisms verify the sender is who she or he claims to be, and that the sender attests to the authenticity of the information.
- *Stamped* to protect against delivery to an incorrect recipient. *Receiver authentication* mechanisms verify the receiver, and *non-repudiation* mechanisms certify receipt.

A variety of Cisco products provide security services. Firewalls are well known, but certainly not the only products. In addition to its firewall offerings, Cisco has a number of other products and services to support AAA, its architecture for:

Authentication -- the process of making a user or system prove that it is who it claims to be. Authentication can be done against addresses, against server/object names, or at the packet level to insure that data is from a legitimate source. It can also be done in a variety of ways from simple pre-shared passwords to the complicated devices or biometrics. For user authentication, the best systems generally combine "something you have and something you know." An example of this would be a fingerprint (which you have) and a memorized password (which you know).

Authorization -- allowing or denying access to specific services or systems (establishing rights or permissions), and

Accounting -- tracking who uses what.

Predominant among these AAA products is CiscoSecure ACS, the Access Control Server, which functions together with the NAS (Network Access Server), or with an external RADIUS or TACACS+ system, to provide access security.

In addition to the AAA services, Cisco has a number of other security products including:

- Cisco Secure Scanner (formerly NetSonar), a tool that can probe for system vulnerabilities, enforce security policies, and provide electronic inventories of network devices and services
- Cisco Intrusion Detection (formerly NetRanger), an intrusion detection system (IDS) that monitors vulnerable systems, detects unauthorized accesses, and terminates these connections. The NetRanger Director can also log intrusion information for future reporting.
- Cisco Netsys Baseline -- this is a great tool for configuration monitoring and management. It can be run against a production network to report on misconfigured routers, or even mismatched router configurations (such as inconsistent IP address masks on routers connected to the same line or network). It can also be used for modeling proposed network changes to test for problems or potential impact, before the changes are actually implemented. Netsys can also report on any configuration changes that have been made, allowing the administrator to troubleshoot network problems or detect unauthorized alterations.

Firewall Components

A "firewall" is generally defined as one or more devices that are installed between networks with different security policies and act to filter traffic passing between the networks. This is done so that the network devices themselves don't all need to be configured securely (an impossible task for most companies). It is usually assumed that one of the networks is the Internet, but a firewall could be installed between individual companies (after a merger or during a business partnership) or between departments within a company.

The simplest router with the least sophisticated IOS code could work very well as a firewall if a company was willing to prohibit almost all outside traffic. (If they were willing to prohibit **all** traffic they wouldn't need a firewall.) Complicated configurations and sophisticated features become necessary as a company wants to permit more and more types of traffic at the same time they want more control over the precise nature of that traffic. The purpose of this paper is to describe some of the most common features of firewalls, show the capabilities of three different Cisco firewall offerings, and help you understand what platform and features you would need to implement security for a particular type of traffic.

Core Capabilities

Generally, a corporate firewall has one or more devices with the following three functions:

Authentication mechanism

The first major component of Internet security is the ability to authenticate users or systems that would access the network. The firewall might not perform the authentication itself, in which case it would have hooks into a separate server that could authenticate users or applications. In addition to authenticating (establishing who or what something is) a firewall may also provide some level of authorization (establishing what the user or system can access) but this is frequently done on a dedicated system or by the individual application.

Packet filtering

This is the most basic function of a firewall but the one most people think of first. It is the ability to selectively reject, or discard, unwanted traffic. Devices that perform packet filtering are frequently referred to as "bastion hosts," and at a minimum, they can screen traffic based on source address, destination address, and TCP/UDP port number for both incoming and outgoing traffic.

Some bastion hosts can provide "stateful" packet inspection -- this refers to a firewall's ability to monitor the "state" of ongoing conversations. It knows, for example, whether a file transfer originated within or outside the network, whether acknowledgments should be transmitted between stations, or whether the sequence numbers on packets transmitted between two stations are incrementing as expected.

Application gateway

These devices are more commonly referred to as "proxies." Their purpose is twofold. They specify which applications can be accessed by internal and external users, and they hide the addresses of internal devices by acting as a "proxy" for internal users accessing external services. This is generally done by terminating the original session and, if access is permitted, creating a second session to the other side.

There are several types of proxies:

- Transport layer proxies using SOCKS/SSL/TLS, or, indeed, manual login
- Application layer gateways aware of application protocols, operations, and embedded addresses
- Traffic-aware application layer gateways, which add the ability to permit or deny certain destinations and limit operations and traffic volumes by destination. Essentially, they add awareness of protocol parameters.
- Content-aware application layer gateways, which add examination of the data carried by protocols. Their functions include checking for viruses and "naughty word" screening for web sites.

Other Security Features of Firewalls

In addition to these major capabilities, there are a number of other popular firewall features:

Address Translation -- This can hide a company's internal network addresses, or allow it to use private addresses, as defined in RFC 1918, on its internal network for traffic that does not pass through a proxy.

Monitoring/Logging -- This allows the security administrator to keep track of suspicious conversations or traffic patterns. Auditing and accountability are fundamental requirements of security.

Support for dial-in access

Encryption -- Data encryption is frequently used to preserve information confidentiality. It can be performed at several different OSI layers, but recently network layer encryption is most often discussed. Encrypted data is a particular problem when providing data security. Since encrypted data cannot be examined by the firewall, the network administrator has to choose between passing without much more than source address verification, or terminating and decrypting the traffic so that it can be examined before it's allowed in.

Firewall Platform Considerations

Finally, there are features on firewalls that are evaluated in much the same way you'd look at any other type of computer:

- Flexibility
- Ease of configuration
- User interface, and
- Interoperability with other systems/standards in use within the company.

IOS Security Features

Frequently for a small office, or for a connection that has limited traffic and an uncomplicated access policy, a router can be used as a firewall. Cisco's IOS software provides a significant amount of protection and has features comparable to those found in a more traditional firewall. These include:

Standard and Extended Access Lists -- Standard lists can be used to control line and port access to the router; extended lists can (and should) be used to filter incoming and outgoing traffic by source address, destination address, and port number. ACLs (Access Control Lists) can also be named, which is useful if you plan to make changes to them frequently. CCIE Access Lists Tutorial.

Timed Access Lists -- Imposing traffic restrictions by time of day can be useful to limit access during non-work hours when the network might not be monitored.

Dynamic Access Lists -- These lists apply statements based on a user's ID and password. This gives the administrator some limited ability to do authentication.

Policy-based routing -- The ability to make routing decisions on something other than the lowest cost path to the destination address (e.g., the source address or type of traffic) gives you much more control over the flow of traffic in the network.

Network Address Translation (NAT) -- Cisco supports both network translation (one address to another) and port translation (multiple addresses to different ports on a single address) to allow you to hide internal (or external) addresses.

Event logging -- You may have specific events, including access list violations, logged to a console terminal or syslog server.

Peer Router Authentication -- Based on the routing protocol used, updates may be authenticated based on the source address of the update.

Tunneling -- Routers can "tunnel" non-IP traffic using GRE, L2F, or L2TP protocols.

In addition to implementing controls on traffic, the router that will function as firewall should be secured itself. Some commonly used techniques for "hardening" a router are discussed next.

Guidelines for Configuring a Firewall

The advantage with a firewall is that it allows you to create a single point of entry for traffic going to or from an external network (e.g. another company, business partner, or the Internet), thus limiting the amount of security you have to provide on each of your internal workstations. A disadvantage of a firewall is exactly that -- that it is a single point of entry. If you have information that you would prefer to keep private, and there is only one device that keeps the rest of the entire world out of that information, you have to be very careful about what you do with that device.

We have discussed some of the IOS features that you may or may not want to include in your firewall. There are some general guidelines, however, that apply to any device attached to an external network:

- Limit all access to the firewall device. Use a RADIUS or TACACS+ server if available, an "enable secret" password if not. Do not use the "enable" command to create a password on the router. The encryption algorithm invoked with the "service password-encryption" is too easy to break. Put passwords on the CON and AUX ports. (Remember that if there is no enable secret password, the password applied to the CON port can be used for privileged access as well.)
- Apply access lists with the "ip access-class" command to restrict who can use the VTY and CON lines. Disable the AUX port if possible.
- If you do choose to support SNMP management on the router, control this with an access list also, limiting it to a few devices only. Use complicated SNMP passwords and, obviously, use different passwords for read-only and read-write passwords.
- Use the "transport input" command to limit VTY sessions protocols. Cisco routers can accept a number of different connection types such as SSH, LAT, MOP, and rlogin in addition to the regular telnet. Only allow ones you are likely to use. Consider using a non-standard port number for telnet access also.
- Set session timers on all router access ports.
- Log access list violations using either the "log" or the newer "log-input" parameters, which provides additional information about the interface and MAC address from which the violation came, on each access list statement. Log system events and access violations (with the "ip accounting access-violations" command) both locally and at a remote location.
- Turn off the CDP at the router, or at a minimum on all external interfaces.
- Disable minor services (such as tcp-small-servers, udp-small-servers, finger, NTP, and http access) unless they are absolutely necessary.
- Use an access list (and the "distribute-list in" command) to restrict the source of routing updates. If the router runs a routing protocol such as EIGRP, authenticate routing updates. Consider not using a routing protocol at all for external interfaces.
- Turn off "ip directed-broadcast" (so that broadcasts are not routed through the network). Turn off source-routing (to prevent source-routed packets from disrupting devices that cannot process them).
- Limit "spoofing" attacks by filtering external traffic that has a source address that's from private address space or internal to the network.
- Disable, or limit, ICMP access coming in to the router. It may be useful (or even necessary) for external users to be able to ping the external interface of the router, but they don't need to ping anything internal to the network.
- You may wish to put a warning banner on the router. Frequently this is done to notify unauthorized users that their traffic may be subject to monitoring. Do not, however, include any other information about the company or the router in this banner. It's not necessary and could be used by an attacker.
- Consider implementing more than the normal user and exec levels of router access. Routers support levels 1 through 15. User level is 1, and exec level is 15, but the "privilege exec level" and "enable password level x" commands can be used to assign specific commands to levels in between. This gives you tighter control over what is done at the router.
- Maintain physical security. Keep the router in a locked room, do not attach a modem to any of the ports unless you have to and if you do attach a modem, make sure it's security (IDs, passwords) is comparable to that of the router. Secure copies of system documentation, including configuration files.

Sample Firewall Router Configuration

The following is an abbreviated sample of the configuration file of a router being used as a firewall. It shows many of the modifications recommended above.

```
RouterA
service password-encryption
no service udp-small-servers
no service tcp-small-servers
no ip helper
```

```

no ip broadcast-address
no ip rcmd rcp-enable
no ip rsh-enable
no ip identd
no proxy-arp
no ip redirects

no ip source-route
no ip http server
no ntp master
logging buffered
logging 10.100.30.15

interface Ethernet0
 ip address 4.1.1.2
 ip access-group from-internet in
 no cdp enable

interface Ethernet1
 ip address 10.100.10.1
 ip access-group from-ss in

interface Ethernet2
 no ip address

access-list 10 permit 10.100.1.0 0.0.0.3
access-list 15 permit 10.100.1.50 255.255.255.255

access-list 101 deny ip 10.0.0.0 0.255.255.255 any
access-list 101 deny ip 127.0.0.0 0.255.255.255 any
access-list 101 deny ip 224.0.0.0 0.255.255.255 any
access-list 101 deny ip 192.168.0.0 0.0.255.255 any
access-list 101 deny ip host 0.0.0.0 any
access-list 101 deny ip host 4.1.1.2 any
access-list 101 permit bgp
access-list 101 permit smtp
access-list 101 permit tcp name_server eq domain
access-list 101 permit udp name_server eq domain
access-list 101 deny ip any any log

enable password level 2 second
privilege exec level 2 show running-config

snmp-server community pswd RO 10
snmp-server trap-authentication
snmp-server enable traps
snmp-server host x.x.x.x password

line aux 0
 transport input none

line con 0
 password 7 abcdefghijklmno

line vty 0 3
 access-class 10 in
 transport input telnet
 login

line vty 4
 access-class 15 in
 transport input telnet
 login

```

Although extended access lists are an excellent -- and sometimes sufficient -- way to control traffic going to or from a protected network, they have the disadvantage of being fixed. Every "permit" statement opens a hole in the firewall for the duration of the time the access list is in place. Dynamic access lists can be used to open security holes selectively and temporarily. We will discuss them next.

IOS Security Feature -- Dynamic ACLs

Dynamic access lists have been available in IOS for a number of releases but are not generally used on purely internal routers. For this reason, many network managers are unfamiliar with this feature.

Dynamic access lists are just that. They are lists whose statements, and therefore access policy, change based on who is using them. This is how they work: an "external" user telnets into the router, presents a valid ID and password, and then logs out. As a result of this interaction, one or more ACL statements are activated for that user, temporarily allowing access for one or more types of traffic.

This has two big advantages over static lists:

- Access through the router can be open temporarily, thus lessening the security risk.
- Access can be based on who someone is rather than the IP address of the device from which the traffic originates (which is very useful for workstations that have either DHCP- or ISP-assigned addresses).

This is the format of the command (italics are variables to be replaced appropriately):

```
access-list number dynamic name [timeout minutes]
      permit|deny protocol any dest_ip dest_mask
```

For example, if user Jane required external access through RouterA to the corporate mainframe (address 10.100.50.5), the appropriate code would be:

```
access-list 110 permit any host 4.1.1.2 eq telnet
access-list 110 dynamic test timeout 30 permit
      tcp any host 10.100.50.5
```

In addition, you must apply the access list to the outside interface and configure the line to which Jane will telnet. In this example we've allowed Jane to telnet on any VTY line; you could restrict that if appropriate.

```
username jane password abcde123

line vty 0 4
  login local
```

(You could also put the "username" command in the line configuration if you wanted the name to apply to that line or lines only.)

You also need to configure a time limit for the dynamic entry. This can be done in one of two ways. The first way is by controlling the idle timer on the VTY line with the command:

```
autocommand access-enable [host] [timeout minutes]
```

Alternatively, you could set an absolute timer on the session in the access list command itself (as shown above). If you choose to do both, make sure that the idle timer is less than the absolute timer.

Some important restrictions:

- Each ACL may have only one dynamic access list, (i.e. you may configure as many as you will like; it will only use the first one).
- The only values changed in a dynamic access list when the list is applied are the source or destination addresses. This is based on whether the list is controlling incoming or outgoing traffic. You may not change any other ACL parameters (such as protocol or port).
- The example shows the user being authenticated locally, but these lists follow Cisco's AAA security paradigm so you may use another device (such as a RADIUS server) for authentication instead.
- Dynamic lists should be used in conjunction with time-based ACLs to restrict when the lists may take effect.

In addition to filtering and controlling traffic between networks, a firewall router may be used to hide the addresses of either internal users, for security, or external users, for security and ease of routing. This can be done using the Network Address Translation feature of IOS.

IOS Security Feature -- Network Address Translation

Network Address Translation (NAT) is one of the few IOS features that's harder to understand than it is to configure. You can actually construct a very functional, adaptable, practical NAT translation in about three or four lines of code. Explaining NAT generally takes about four paragraphs, a few very good drawings, and one or two clever analogies. Because this is a security paper, and not a NAT paper, this will only be an overview of some of the ways to translate addresses. NAT is a very powerful IOS feature, however, and it's worth taking the time to learn the process thoroughly.

NAT, simply put, is the ability to take the source or destination address of traffic entering a router and convert it to another address as it leaves the router. In addition, the router will track the "cloaked" traffic and translate appropriately when it returns. It can be used to:

- Hide internal addresses
- Allow administrators to use RFC 1918, or valid but unassigned IP, addresses internally
- Assign the same address (usually the outside, advertised, interface on the router) to all outgoing users (This is also called Port Address Translation).
- Provide TCP load sharing between hosts.

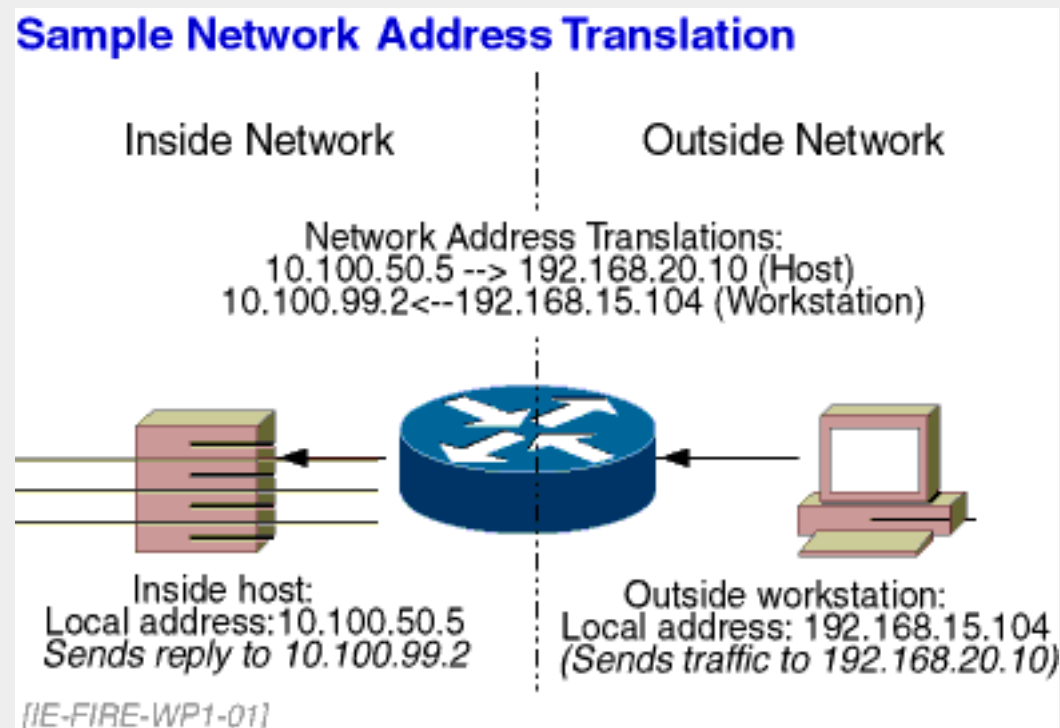
There are a number of different features and permutations possible, but essentially this is it. Unfortunately, in practice, it seems a little more complicated and takes a while to get used to. In addition, NAT has limitations:

- It requires IOS 11.2 or higher.
- The process of translating addresses and tracking the translations is memory and CPU-intensive.
- With some exceptions, it cannot be used with applications that imbed the IP address in the data portion of the IP packet.
- It does not work properly with IP multicast traffic, with DNS zone transfers, or with routing table updates.

Nonetheless, NAT can be useful in a number of situations. For example, the corporation that owns RouterA (described above) forms a temporary business partnership with a consulting firm. For the duration of the relationship, the consulting firm can have a direct link to the corporation's network through which a handful of staff will access a mainframe (at address 10.100.50.5).

The problem is that their network uses 192.168.0.0 addresses, and as you saw in the configuration file, our network uses 10.100.0.0 addresses. Both companies have large intranets with dozens of routers, and neither wants to configure them to route traffic from the other network. This is a situation where NAT can be very useful.

This scenario actually requires two different types of translation. The mainframe will have to have a static translation. Since their software will have an embedded destination address, it must be unchanging. Their users, however, can draw from a pool of addresses each time they start a mainframe session. They don't need to retain their specific addresses once the sessions terminate.



The process of configuring both translations is fairly simple:

1. Identify the "inside" NAT interface (this doesn't actually have to be the interface that's attached to the internal, or more internal, network on the router, but it does make things easier if you're consistent. We're going to label Ethernet1 the "internal" interface.)
2. Identify the "outside" NAT interface. (This will be Ethernet 2 in our example.)
3. For the static translation, write a statement that tells the router:
 - o Which interface will receive incoming traffic from this device (the internal interface in our example)
 - o The address of the device to be translated (10.100.50.5)
 - o The address to translate the device into (192.168.20.10) supplied by the consulting firm's IT staff
 - o The fact that this is a static translation

For the dynamic (user) translation, there are a few more steps:

4. Create an access list that defines which addresses will be translated (access list 1, which will allow anything from 192.168.0.0 in our example)
5. Identify a "pool" of addresses to which the external addresses will be translated. (Each incoming foreign address would get one pool address assigned to it. In our example, the pool will be called "consulting-users.")
6. Write a statement (similar to the one for the static translation) that ties all of this together by telling the router:
 - o Which interface will receive traffic from these addresses (Ethernet 2, the "outside" interface)
 - o Which access list specifies the "legitimate" addresses for translation (those passing access list 1)
 - o From which pool of addresses they will take an "internal" address (consulting-users).

So, assuming that we've connected the consulting firm's router to Ethernet 2 on RouterA, the abbreviated configuration file for the router would look like this:

```
interface Ethernet1
 ip address 10.100.10.1 255.255.255.0
 ip access-group from-ss in
 ip nat inside

interface Ethernet2
 ip address 192.168.20.2
 ip address 10.100.99.1 255.255.255.0 secondary
 ip access-group from-consulting in
 ip nat outside

ip nat inside source static 10.100.50.5 192.168.20.10
ip nat pool consulting-users 10.100.99.2 10.100.99.50
 netmask 255.255.255.0

ip nat outside source list 1 pool consulting-users

access-list 1 permit 192.168.0.0 0.0.255.255
```

Once this is implemented, the external users should be able to use the mainframe by directing their software to 192.168.20.10. Routing these users across the company's network should be easy -- they all come from a "network" that is attached, via a secondary address, to Ethernet2 on RouterA. If RouterA runs a dynamic routing protocol that includes the 10.100.x.x networks, no other changes will be necessary.

While the configuration itself is fairly straightforward, the process of troubleshooting and maintaining routers with NAT can be rather complicated. This is because one has to remember that a device's address will be different depending on the point in the network from which it is viewed it (in the same way that you have to be aware that tunneled traffic is addressed differently within the tunnel than it is before it enters or after it leaves).

As stated before, users at the consulting company always refer to the mainframe by the address 192.168.20.10 and the first session that is started on it is from user 10.100.99.1. The second session will probably come from 10.100.99.2, and so on. Traces, event logs, and other sources of network information, therefore, always have to be interpreted in this context.

Address translation has been a feature of the IOS code since the 11.3 releases. It is useful as a way of hiding internal addresses, but can also be adapted for a number of other purposes, including:

- Assigning several devices the same address (for application fail-over or load sharing)
- Controlling access (because you can control the source address of NAT'd devices and then use ACLs on those addresses)
- Manipulating routing (with policy-based routes).

NAT, as well as Dynamic ACLs and the range of other IOS features previously listed, can allow a router to be used as an effective firewall for many situations. For more complicated traffic control, however, Cisco has a specialized version of router software called the "Firewall Feature Set."

The IOS Firewall Feature Set

Starting with version 11.2(11)P, Cisco has offered an enhanced security version of IOS code called the "Firewall Feature Set." This code was designed to provide many of the features of more advanced firewall products on a multi-protocol router. Cisco felt that this provided more flexibility because it combined the functions of router and firewall in one device. They felt that it was easier to use since the configuration commands would be familiar to someone who was already familiar with the IOS command line interface.

This code is available for certain router models only, as shown in the chart below:

IOS Firewall Feature Set - Supported Platforms and Releases*

| Software (minimum version) | Routers Supported |
|----------------------------|--|
| 11.2(11)P | 1600, 2500 |
| 11.3(3)T | 1600, 2500 |
| 12.0 | 1600, 2500 |
| 12.0(1)T | 1600, 2500, 2600, 3600 |
| 12.0(1)XA | 1720 |
| 12.0(2)T | 1600, 1720, 2500, 2600, 3600 |
| 12.0(3)T | 1600, 1720, 2500, 2600, 3600, 7200 |
| 12.0(4)T | 800,uBR904, 1600, 1720, 2500, 2600, 3600, 7200 |
| 12.0(4)XA | 7100 |
| 12.0(5)T | 800,1600, 1720, 2500, 2600, 3600, 7100, 7200 |

* adapted from Cisco Product Catalog, June 2000

There are three images of Firewall Feature Set code -- IP, Desktop, and Enterprise. Each of these includes the "plain-vanilla" firewall version as well as versions that have IPSec with DES and 3DES encryption.

Initially, the Firewall Feature Set included five additional features:

Context-based access controls (CBAC) -- This is Cisco's implementation of what is commonly referred to as "stateful" packet inspection. The router can make filtering decisions on the basis of application layer information about the "state" of a conversation between two devices (such as whether the traffic was initiated internally, or needs to open a new data channel). This facility is also used for Java applet blocking and preventing DoS attacks (described below). We will discuss it in more detail later.

Java applet blocking -- This allows HTTP traffic to be filtered in several ways: to reject applets not imbedded in archive or compressed files, to reject applets from specific source IP addresses, and to filter applets with standard ACLs.

Denial of Service attack prevention/detection -- this is designed to prevent two common DoS attacks. The SYN attack is designed to overload system resources by creating high numbers of incoming connections that are left open, thus taxing the system and blocking legitimate users' access to resources. The router monitors incoming connections, detects unusually high numbers, and drops old half-open ones. The packet injection attack

is designed to send traffic to an internal device, sometimes by mimicking packets from an established session. The router monitors packet sequence numbers of ongoing communications sessions and drops suspicious traffic.

Real-time alerts -- These are generated to a system console for potential DoS and SMTP attacks, as well as for denied traffic such as Java applets.

Enhanced audit trails -- Information such as date and time, source, destination, port number and bytes transmitted, is logged for all connections going through the router.

In the 12.0(5) and 12.0(6) versions, the code was augmented with other features varying by router model. The 800, uBR900, 1600, and 2500 series routers also supported:

Port-to-Application Mapping (PAM) -- This is a common feature of application proxies. It allows the router to assign non-standard port numbers to be used for applications. The router maintains a table of well-defined (such as 21 for Telnet and 80 for HTTP) and user-defined (for non-standard application ports) that the router can then use to apply -- by host, if required -- a non-standard port or range of ports to a service or application. This is used in conjunction with CBAC to monitor ongoing conversations on non-standard ports.

Configurable alerts and audit trails

SMTP attack prevention/detection -- This is done by limiting incoming SMTP traffic to using standard, documented commands and rejecting traffic that has unrecognized or undocumented commands

Support for MS Netshow.

Model 1720, 2600, 3600, 7100, and 7200 routers had all of the above features plus:

Intrusion detection -- This feature allows the router to act as an IDS (intrusion detection system) and monitor traffic for signs of incoming attacks. The IOS code has signatures for the 59 most common attacks and can be set to send an alarm, reject the traffic, and/or reset the connection when a series of packets match one of the known signatures. This is somewhat analogous to virus detection, however, in that it must be configured carefully because the process of inspecting packets -- entire packets -- and storing a sufficient number of packets from each connection to match a known intrusion signature can have a large performance and memory impact on the router.

Dynamic authentication/authorization proxy support -- This supports user authentication and per-user authorization policies for HTTP connections. This is not a trivial service to configure. The user authentication is done via a RADIUS, or TACACS+, server and the user-specific ACLs must be maintained on an AAA server. This allows a user to be authenticated independent from his IP address, so this works well in situations where users are on workstations with dynamically assigned addresses.

The PIX Firewall

For more complicated security requirements, or for connections to larger networks, Cisco offers the PIX Firewall. Although there are several models ranging in size and price, the PIX is generally used in situations where a router with IOS (of any flavor) would be too slow or too inflexible.

The features of IOS -- particularly the Firewall Feature Set code -- are similar to the PIX, so the functionality of the two devices are similar. It might be useful to differentiate the two by saying that a router is built to do routing and can be adapted to do sophisticated filtering and monitoring, while a PIX was built as a firewall -- a very fast, effective firewall -- that can do only minimal router functions.

Some specific differences between routers and PIX boxes are:

- Routers can be configured to do address translation, if necessary. A PIX will not allow traffic to pass to a less secure (i.e., external) interface unless it is translated, even if the address is translated into itself. This is because the PIX tracks connections by entries in an "xlate" table, and the table is populated initially by the translated addresses of outgoing devices.
- "Stateful" packet inspection has to be configured on a router using context-based access controls. This application level monitoring happens automatically on a PIX. It is referred to as the PIX's "Adaptive Security Algorithm."
- Unless specifically configured not to do so, PIX Firewalls automatically block all traffic attempting to go from an external (lower security) to an internal (higher security) interface.
- A PIX is inherently more secure than an IOS-based device. Most of the recommendations mentioned in the section on general guidelines for configuring a router as a firewall exist by default on a PIX. You don't have to "harden" a PIX.
- A PIX can process only TCP/IP traffic. It has no multi-protocol support.
- A PIX can have only serial or Ethernet (10/100 on the newer units) interfaces. In addition, they generally have only two interfaces, though you can purchase a four-port Ethernet adapter.
- Each port on a PIX has to be assigned a name and a security level (from 1 - 100). Most PIX rules/defaults about passing traffic are applied based on the security level of the source port and the security level of the destination port. An example is the item listed above that says that traffic is automatically blocked if it is passing from a lower security level interface to one that is higher. There are similar rules for other PIX features. (A corollary of this is that traffic cannot pass between PIX interfaces that have been configured to have the same security level.)
- PIX Firewalls cannot participate in a dynamic routing protocol; they are configured with static routes only.
- The commands used to interact with PIX Firewalls are similar to those used for routers using IOS version 10.3. (So you can't use "wr" to save the configuration file, you have to use "wr mem".) This can sometimes be very handy and it can sometimes drive you nuts.

The actual configuration commands used on a PIX are also somewhat similar to those used for a router. This, for example, is a sample of what part of the configuration of RouterA might look like if it were a PIX Firewall:

```
: Saved  
:
```

```

PIX Version 5.0 (1)
nameif ethernet0 outside security0
nameif ethernet1 inside security100
enable password xbpwKLIiL5tlz encrypted
password hWNeazAk0pDczvVyU encrypted
hostname RouterA
fixup protocol ftp 21
fixup protocol http 80
no fixup protocol smtp 25
no logging timestamp
no logging standby
no logging monitor
no logging console
logging trap notifications
logging queue 512
logging host inside 10.100.1.50
interface ethernet0 100basetx
interface ethernet1 100basetx
mtu outside 1500
mtu inside 1500
ip address outside x.x.x.x 255.255.255.252
ip address inside x.x.x.x 255.255.255.192
no failover
failover timeout 0:00:00
arp timeout 14400
nat (inside) 0 0.0.0.0 0.0.0.0 0
static (inside, outside) 204.32.26.0 204.32.16.0
    netmask 255.255.255.192
conduit permit tcp host 10.100.1.50 eq telnet host
    204.32.26.1
conduit permit icmp host x.x.x.x any eq echo
conduit permit icmp host {ext. interface} eq
    time-exceeded
conduit permit icmp host {ext interface} eq unreachable
.....(other conduit statements as needed)
route outside 0.0.0.0 0.0.0.0 {ISP interface addr} 1
route inside 10.100.0.0 255.255.0.0
timeout xlate 3:00:00 conn 1:00:00 half-closed
    0:10:00 udp 0:02:00
aaa-server TACACS+ protocol tacacs+
snmp-server host inside 10.100.1.49
telnet 10.100.1.35 255.255.255.192 inside
telnet timeout 30
Cryptochecksum: 90672abd55787fdbbb72a93

```

There are a few things worth noticing in this example:

- There is a NAT configuration -- the "static" command is used for devices coming through lower security interfaces, whereas the "nat" command is used for devices coming from a higher security level. In this configuration, the inside device addresses are being translated into themselves.
- "Conduit" statements are used to allow traffic from lower security level interfaces to enter a network behind a higher security level interface. Though it looks somewhat like an access list statement, in a conduit statement the destination address is written first and the normal network mask, not a wildcard mask, is used to specify the subnet. The 5.x version of PIX code actually supports an "access-list" command, but still uses the network mask instead of a wildcard mask.
- Many security holes in routers are closed on a PIX. You have to specifically identify which devices can telnet to the unit. Also, you must specify translation, connection, and half-open connection timeouts, for example.
- There are no "permit" statements for traffic going from the "inside" to the "outside" interfaces. This traffic is permitted by default. Stateful inspection of every session happens automatically.

Conclusion

This paper should have given you a general idea of some of the features commonly found in firewalls and the extent to which Cisco's IOS, IOS Firewall Feature Set, and PIX Firewall products can be used to implement these features. Selecting and configuring an appropriate firewall, or firewalls, will be based on a careful analysis of the data security needs of the company, the nature of the "external" network against which the firewall is supposed to defend, and the type of traffic that needs to be permitted between the networks. This is not a simple task, and there are no clear-cut rules for selecting one device or one feature over another. There are a number of excellent books available on computer security; some of them are listed as references at the end of this paper. One or more of them might be helpful in making these decisions.

References

Chapman, D. Brent, and Zwicky, Elizabeth. *Building Internet Firewalls*, O'Reilly and Associates.

Cheswick, and Bellovin, S. *Firewalls and Internet Security: Foiling the Wily Hacker*. Addison-Wesley.

Cisco Technical Tip "Improving Security on Cisco Routers" <http://www.cisco.com/warp/public/707/21.html>. Posted: Monday July 26, 1999.

Shockley, P. "IOS Security Features" Cisco Systems Federal Technical Seminar Series.

NSA Aqua Book at <http://www.radium.ncsc.mil/tpep/library/rainbow/NCSC-TG-010.txt>

NSA Orange Book at <http://www.radium.ncsc.mil/tpep/library/rainbow/5200.28-STD.txt>

NSA Red Book at <http://www.radium.ncsc.mil/tpep/library/rainbow/NCSC-TG-005.txt>.

RFC 2827 Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing. P. Ferguson, D. Senie. May 2000.

RFC 1579 Firewall-Friendly FTP. S. Bellovin. February 1994.

RFC 1812 Requirements for IP Version 4 Routers. F. Baker. June 1995.

RFC 2644 Changing the Default for Directed Broadcasts in Routers. D Senie. August 1999.

Wack, John P. and Carnahan, Lisa J., *Keeping Your Site Comfortably Secure: An Introduction to Internet Firewalls*, NIST Special Publication 800-10, U.S. Department of Commerce, National Institute of Standards and Technology.

Firewalls Lab

[Tutorial](#)
[Lab Abstract](#)
[Lab Scenario](#)

Dynamic Access List/Firewalls - In this lab, we will be using a dynamic access list statement to allow one router to ping another. Though this is probably not the type of thing you would do on a production network, it does illustrate how dynamic ACLs work.