# Configuring AppleTalk

**Note** Effective with Cisco IOS Release 15.0(1)M, this feature is not available in Cisco IOS software.

This chapter describes how to configure AppleTalk and provides configuration examples. For a complete description of the AppleTalk commands mentioned in this chapter, refer to the *Cisco IOS AppleTalk Command Reference* publication. To locate documentation for other commands that appear in this chapter, use the command reference master index or search online.

For the latest feature information and caveats, see the release notes for your platform and software release. Additionally, use Cisco Feature Navigator to find information about feature, platform, and software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

# AppleTalk Phases

The AppleTalk network architecture has the following two phases:

- AppleTalk Phase 1
- AppleTalk Phase 2

## AppleTalk Phase 1

*AppleTalk Phase 1* is the initial implementation of AppleTalk and is designed for logical workgroups. AppleTalk Phase 1 supports a single physical network that can have one network number and be in one zone. This network can have up to 254 devices, which can consist of 127 end nodes and 127 servers.

## AppleTalk Phase 2

*AppleTalk Phase 2* is an enhancement to AppleTalk Phase 1 and is designed for larger networks and has improved routing capabilities. It supports multiple logical networks on a single physical network and multiple logical networks in a given zone, which means that one cable segment can have multiple

**Americas Headquarters:**
**Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA**

network numbers. Each logical network in Phase 2 can support up to 253 devices, with no restrictions on the type of devices (end nodes or servers). Also, in AppleTalk Phase 2, a network can be in more than one zone.

## Types of AppleTalk Phase 2 Networks

AppleTalk Phase 2 distinguishes between two types of networks based on their media-level encapsulation and cable addressing methods. The two types of networks are as follows:

- Nonextended
- Extended

Table 1 compares the attributes of nonextended and extended networks.

*Table 1 Comparison of Nonextended and Extended Networks*

| Attribute | Nonextended | Extended |
|---|---|---|
| Media-level encapsulation method | Encapsulation of the 3-byte LocalTalk packet in an Ethernet frame | ISO-type encapsulations only (that is, no encapsulation of the 3-byte LocalTalk packets) |
| Physical media that supports media-level encapsulation methods | LocalTalk | All physical media except LocalTalk |
| Node addressing method | Each node number is unique | Each *network.node* combination is unique |
| Cable addressing method | A single number per cable | A number range corresponding to one or more logical networks |

Nonextended networks were the sole network type defined in AppleTalk Phase 1. You can consider AppleTalk Phase 1 networks to be nonextended networks.

You can consider AppleTalk Phase 2 networks to be extended networks.

Table 2 compares the capabilities of AppleTalk Phase 1 and Phase 2.

*Table 2 Comparison of AppleTalk Phase 1 and Phase 2*

| Capability | AppleTalk Phase 1 | AppleTalk Phase 2 |
|---|---|---|
| **Networks, nodes, and zones** | | |
| Number of logical networks (cable segments) | 1 | 65,279[1] |
| Maximum number of devices | 254[2] | 253[3] |
| Maximum number of end nodes | 127 | Does not apply[4] |
| Maximum number of servers | 127 | Does not apply |
| Number of zones in which a network can be | 1[5] | 1 (nonextended)<br>255 (extended) |

*Table 2 Comparison of AppleTalk Phase 1 and Phase 2 (Continued)*

| Capability | AppleTalk Phase 1 | AppleTalk Phase 2 |
|---|---|---|
| **Media-level encapsulation** | | |
| Nonextended network | Does not apply | Yes |
| Extended network | Does not apply | Yes |
| Cable addressing | Does not apply; uses network numbers | Single network number (nonextended)<br>Cable range of 1 or more (extended) |

1. The 65,279 value is per AppleTalk specifications.
2. The node addresses 0 and 255 are reserved.
3. The node addresses 0, 254, and 255 are reserved.
4. There is no restriction on the types of devices. There can be a total of 253 end nodes and servers.
5. In terms of zones, an AppleTalk Phase 1 network can be thought of as a nonextended AppleTalk Phase 2 network.

Routers running Cisco IOS software Release 8.2 or later support AppleTalk Phase 1 and Phase 2.

# AppleTalk Addresses

An AppleTalk *address* consists of a network number and a node number expressed in decimal in the format *network.node*.

## Network Numbers

The *network number* identifies a network, or cable segment. A *network* is a single logical cable. Although the logical cable is frequently a single physical cable, bridges and routers can interconnect several physical cables.

The network number is a 16-bit decimal number that must be unique throughout the entire AppleTalk internetwork.

In AppleTalk Phase 1, networks are identified by a single network number that corresponds to a physical network. In AppleTalk Phase 1, the network number 0 is reserved.

In AppleTalk Phase 2, networks are identified by a cable range that corresponds to one or more logical networks. In Phase 2, a single cable can have multiple network numbers.

A cable range is either one network number or a contiguous sequence of several network numbers in the format *start–end*. For example, the cable range 4096–4096 identifies a logical network that has a single network number, and the cable range 10–12 identifies a logical network that spans three network numbers.

In AppleTalk Phase 2, the network number 0 is reserved.

## Node Numbers

The *node number* identifies the node, which is any device connected to the AppleTalk network. The node number is an 8-bit decimal number that must be unique on that network.

In AppleTalk Phase 1, node numbers 1 through 127 are for user nodes, node numbers 128 through 254 are for servers, and node numbers 0 and 255 are reserved.

In AppleTalk Phase 2, you can use node numbers 1 through 253 for any nodes attached to the network. Node numbers 0, 254, and 255 are reserved.

## AppleTalk Address Example

The following is an example of an AppleTalk network address:

```
3.45
```

In this example, the network number is 3 and the node number is 45. You enter both numbers in decimal. Cisco IOS software also displays them in decimal.

# AppleTalk Zones

A *zone* is a logical group of networks. The networks in a zone can be contiguous or noncontiguous. A zone is identified by a zone name, which can be up to 32 characters long. The zone name can include standard characters and AppleTalk special characters. To include a special character, type a colon followed by two hexadecimal characters that represent the special character in the Macintosh character set.

An AppleTalk Phase 1 network can have only one zone.

In AppleTalk Phase 2, an extended network can have up to 255 zones; a nonextended network can have only 1 zone.

# Configuration Guidelines and Compatibility Rules

AppleTalk Phase 1 and AppleTalk Phase 2 networks are incompatible and cannot run simultaneously on the same internetwork. As a result, all routers in an internetwork must support AppleTalk Phase 2 before the network can use Phase 2 routing.

If your internetwork has a combination of AppleTalk Phase 1 and Phase 2 routers, you must observe the following configuration guidelines. If you do not follow these guidelines, unpredictable behavior might result. Note, however, that you do not need to upgrade all end nodes to use the features provided by our AppleTalk enhancements.

- The cable range must be one (for example, 23–23).
- Each AppleTalk network can be a member of only one zone.

When using Cisco routers with implementations of AppleTalk by other vendors, follow these guidelines:

- For a Macintosh with an Ethernet card to support extended AppleTalk, the Macintosh must be running EtherTalk Version 2.0 or later. This restriction does not apply to Macintoshes with only LocalTalk interfaces.

- Shiva FastPath routers must run K-Star Version 8.0 or later, and must be explicitly configured for extended AppleTalk.
- Apple Internet Router software Version 2.0 supports a transition mode for translation between nonextended AppleTalk and extended AppleTalk on the same network. Transition mode requires the Apple upgrade utility and a special patch file from Apple.

# AppleTalk Configuration Task List

To configure AppleTalk routing, perform the tasks in the following sections:

- Configuring AppleTalk Routing (Required)
- Controlling Access to AppleTalk Networks (Optional)
- Configuring the Name Display Facility (Optional)
- Setting Up Special Configurations (Optional)
- Configuring AppleTalk Control Protocol for PPP (Optional)
- Tuning AppleTalk Network Performance (Optional)
- Configuring AppleTalk Interenterprise Routing (Optional)
- Configuring AppleTalk over WANs (Optional)
- Configuring AppleTalk Between LANs (Optional)
- Configuring AppleTalk Between VLANs (Optional)
- Monitoring and Maintaining the AppleTalk Network (Optional)

See the "AppleTalk Configuration Examples" section at the end of this chapter for configuration examples.

## Configuring AppleTalk Routing

You configure AppleTalk routing by first enabling it on the router and then configuring it on each interface.

To configure the AppleTalk routing protocol, perform the tasks in the following sections. The first two tasks are required; the rest are optional.

- Enabling AppleTalk Routing (Required)
- Configuring an Interface for AppleTalk (Required)
- Selecting an AppleTalk Routing Protocol (Optional)
- Configuring Transition Mode (Optional)
- Enabling Concurrent Routing and Bridging (Optional)
- Configuring Integrated Routing and Bridging (Optional)

### Enabling AppleTalk Routing

To enable AppleTalk routing, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk routing** | Enables AppleTalk routing. |

The **appletalk routing** command without any keywords or arguments enables AppleTalk routing using the Routing Table Maintenance Protocol (RTMP) routing protocol.

For an example of how to enable AppleTalk routing, see the "Extended AppleTalk Network Example" section at the end of this chapter.

## Configuring an Interface for AppleTalk

You configure an interface for AppleTalk by assigning an AppleTalk address or cable range to the interface, and then assigning one or more zone names to the interface. You can perform these tasks either manually or dynamically.

### Manually Configuring an Interface

You can manually configure an interface for nonextended AppleTalk or extended AppleTalk routing.

#### Configuring for Nonextended AppleTalk Routing

To manually configure an interface for nonextended AppleTalk routing, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | Router(config-if)# **appletalk address** *network***.***node* | Assigns an AppleTalk address to the interface. |
| **Step 2** | Router(config-if)# **appletalk zone** *zone-name* | Assigns a zone name to the interface. |

After you assign the address and zone names, the interface will attempt to verify them with another operational router on the connected network. If there are any discrepancies, the interface will not become operational. If there are no neighboring operational routers, the device will assume the configuration of the interface is correct, and the interface will become operational.

For an example of how to configure an interface for nonextended AppleTalk routing, see the "Nonextended AppleTalk Network Example" section at the end of this chapter.

#### Configuring for Extended AppleTalk Routing

To manually configure an interface for extended AppleTalk routing, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | Router(config-if)# **appletalk cable-range** *cable-range* [*network***.***node*] | Assigns a cable range to an interface. |
| **Step 2** | Router(config-if)# **appletalk zone** *zone-name* | Assigns a zone name to the interface. |

You can assign more than one zone name to a cable range. If you do so, the first name you assign is considered to be the default zone. You can define up to 255 zones.

For an example of how to configure an interface for extended AppleTalk routing, see the "Extended AppleTalk Network Example" section at the end of this chapter.

## Dynamically Configuring an Interface

If a nonextended or an extended interface is connected to a network that has at least one other operational AppleTalk router, you can dynamically configure the interface using *discovery mode*. In discovery mode, an interface acquires information about the attached network from an operational router and then uses this information to configure itself.

### Benefits

Using discovery mode to configure interfaces saves time if the network numbers, cable ranges, or zone names change. If any of these changes occur, you must make the changes on only one seed router on each network.

Discovery mode is useful when you are changing a network configuration or when you are adding a router to an existing network.

### Restrictions

If there is no operational router on the attached network, you must manually configure the interface as described in the previous sections. Also, if a discovery mode interface is restarted, another operational router must be present before the interface will become operational.

Discovery mode does not run over serial lines.

**Caution** Do not enable discovery mode on all routers on a network. If you do so and all the devices restart simultaneously (for example, after a power failure), the network will be inaccessible until you manually configure at least one router.

### Seed Router Starting Sequence

A nondiscovery-mode interface (also called a *seed router*) starts up as follows:

1. The seed router acquires its configuration from memory.
2. If the stored configuration is not completely specified when you assign an AppleTalk address to an interface on which you assign a cable range and a zone name, the interface will not start up.
3. If the stored configuration is completely specified, the interface attempts to verify the stored configuration with another router on the attached network. If any discrepancy exists, the interface will not start up.
4. If there are no neighboring operational routers, the device will assume the X stored configuration of the interface is correct, and the interface will become operational.

### Response to Configuration Queries

Using discovery mode does not affect the ability of an interface to respond to configuration queries from other routers on the connected network once the interface becomes operational.

### Dynamically Configuring a Nonextended Interface

You can activate discovery mode on a nonextended interface in one of two ways, depending on whether you know the network number of the attached network.

In the first method, you immediately place the interface into discovery mode by specifying an AppleTalk address of 0.0. Use this method when you do not know the network number of the attached network. To activate discovery mode for this method, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk address 0.0` | Places the interface into discovery mode by assigning it the AppleTalk address 0.0. |

For an example of how to configure discovery mode using this method, see the "Nonextended Network in Discovery Mode Example" section at the end of this chapter.

For the second method, you first assign an address to the interface and then explicitly enable discovery mode. Use this method when you know the network number of the attached network. Note, however, that you are not required to use this method when you know the network number. To activate discovery mode for this method, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# appletalk address network.node` | Assigns an AppleTalk address to the interface. |
| **Step 2** | `Router(config-if)# appletalk discovery` | Places the interface into discovery mode. |

### Dynamically Configuring an Extended Interface

You can activate discovery mode on an extended interface in one of two ways, depending on whether you know the cable range of the attached network.

In the first method, you immediately place the interface into discovery mode by specifying a cable range of 0–0. Use this method when you do not know the network number of the attached network. To activate discovery mode for this method, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk cable-range 0-0` | Places the interface into discovery mode by assigning it the cable range 0–0. |

In the second method, you first assign cable ranges and then explicitly enable discovery mode. Use this method when you know the cable range of the attached network. Note, however, that you are not required to use this method if you know the cable range. To activate discovery mode for this method, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# appletalk cable-range cable-range [network.node]` | Assigns an AppleTalk address to the interface. |
| **Step 2** | `Router(config-if)# appletalk discovery` | Places the interface into discovery mode. |

## Selecting an AppleTalk Routing Protocol

Once you configure AppleTalk on an interface, you can select a routing protocol for the interface. You can enable the RTMP routing protocol on any interface. You can also enable the AppleTalk Update-Based Routing Protocol (AURP) on a tunnel interface.

With the **appletalk protocol** command, you can enable some AppleTalk interfaces to use RTMP and others to use AURP as required by your network topology.

To select an AppleTalk routing protocol for an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk protocol {aurp | rtmp}` | Creates an AppleTalk routing process. |

This command is optional. If you do not select a routing protocol for an interface, Cisco IOS software uses RTMP by default.

## Configuring Transition Mode

The Cisco IOS software can route packets between extended and nonextended AppleTalk networks that coexist on the same cable. This type of routing is referred to as *transition mode*.

To use transition mode, you must have two router ports connected to the same physical cable. One port is configured as a nonextended AppleTalk network, and the other port is configured as an extended AppleTalk network. Each port must have a unique network number, because you are routing between two separate AppleTalk networks: the extended network and the nonextended network.

To configure transition mode, you must have two ports on the same router that are connected to the same physical cable. To configure one port as a nonextended AppleTalk network, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# appletalk address network.node` | Assigns an AppleTalk address to the interface. |
| **Step 2** | `Router(config-if)# appletalk zone zone-name` | Assigns a zone name to the interface. |

To configure the second port as an extended AppleTalk network, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# appletalk cable-range cable-range [network.node]` | Assigns an AppleTalk cable range to the interface. |
| **Step 2** | `Router(config-if)# appletalk zone zone-name` | Assigns a zone name to the interface. |

When you enter interface configuration mode, the type of interface must be the same for both ports (for example, both could be Ethernet) and the interface number must be different (for example, 0 and 1).

For an example of how to configure transition mode, see the "Transition Mode Example" section at the end of this chapter.

### Enabling Concurrent Routing and Bridging

You can route AppleTalk on some interfaces and transparently bridge it on other interfaces simultaneously. To enable this type of routing, you must enable concurrent routing and bridging.

To enable concurrent routing and bridging, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# bridge crb` | Enables concurrent routing and bridging. |

### Configuring Integrated Routing and Bridging

Integrated routing and bridging (IRB) enables a user to route AppleTalk traffic between routed interfaces and bridge groups, or route AppleTalk traffic between bridge groups. Specifically, local or unroutable traffic is bridged among the bridged interfaces in the same bridge group, while routable traffic is routed to other routed interfaces or bridge groups.

Using IRB, you can do the following:

- Switch packets from a bridged interface to a routed interface
- Switch packets from a routed interface to a bridged interface
- Switch packets within the same bridge group

For more information about configuring integrated routing and bridging, refer to the “Configuring Transparent Bridging” chapter in the *Cisco IOS Bridging and IBM Networking Configuration Guide*.

## Controlling Access to AppleTalk Networks

An *access list* is a list of AppleTalk network numbers, zones, or Name Binding Protocol (NBP) named entities that is maintained by the Cisco IOS software and used to control access to or from specific zones, networks, and NBP named entities.

### Types of Access Lists

The software supports the following two general types of AppleTalk access lists:

- AppleTalk-style access lists, which are based on AppleTalk zones or NBP named entities
- IP-style access lists, which are based on network numbers

#### AppleTalk-Style Access Lists

AppleTalk-style access lists regulate the internetwork using zone names and NBP named entities. The main advantage of AppleTalk-style access lists is that they allow you to define access regardless of the existing network topology or any changes in future topologies—because they are based on zones and NBP named entities. A zone access list is essentially a dynamic list of network numbers. The user specifies a zone name, but the effect is as if the user had specified all the network numbers belonging to that zone. An NBP named entity access list provides a means of controlling access at the network entity level.

#### Using Zone Names

Zone names and NBP named entities are good control points because they allow for network-level abstractions that users can access.

You can express zone names either explicitly or by using generalized-argument keywords. Thus, using AppleTalk zone name access lists simplifies network management and allows for greater flexibility when adding segments, because reconfiguration requirements are minimal. Using AppleTalk zone name access lists allows you to manage and control whole sections of the network.

#### Using NBP Named Entities

NBP named entities allow you to control access at the object level. Using NBP named entities, you can permit or deny NBP packets from a class of objects based on the **type** portion of the NBP tuple name, from a particular NBP named entity based on the **object** portion of the NBP tuple name, or from all NBP named entities within a particular area based on the **zone** portion of the NBP tuple name. You can fully or partially qualify an NBP tuple name to refine the access control by specifying one, two, or three parts of the NBP name tuple as separate access list entries tied together by the same sequence number.

### IP-Style Access Lists

IP-style access lists control network access based on network numbers. This feature can be useful in defining access lists that control the disposition of networks that overlap, are contained by, or exactly match a specific network number range.

Additionally, you can use IP-style access lists to resolve conflicting network numbers. You can use an access list to restrict the network numbers and zones that a department can advertise, thereby limiting advertisement to an authorized set of networks. AppleTalk-style access lists are typically insufficient for this purpose.

In general, however, using IP-style access lists is not recommended because the controls are not optimal; they ignore the logical mapping provided by AppleTalk zones. One problem with IP-style access lists is that when you add networks to a zone, you must reconfigure each secure router. Another problem is that, because anyone can add network segments (for example, when one group of users gets a LaserWriter and installs a Cayman GatorBox, creating a new network segment), the potential for confusion and misconfiguration is substantial.

### Combining AppleTalk-Style and IP-Style Entries

You can combine zone, network, and NBP named entity entries in a single access list. Cisco IOS software performs NBP filtering independently on only NBP packets. The software applies network filtering in conjunction with zone filtering. However, for optimal performance, access lists should not include both zones (AppleTalk-style) and numeric network (IP-style) entries.

Because the Cisco IOS software applies network filtering and zone filtering simultaneously, be sure to add the appropriate **access-list permit other-access** or **access-list permit additional-zones** statement to the end of the access list when using only one type of filtering. For example, suppose you want to deny only zone Z. You do not want any network filtering, but the software by default automatically includes an **access-list deny other-access** entry at the end of each access list. You must then create an access list that explicitly permits access of all networks. Therefore, the access list for this example would have an **access-list deny zone Z** entry to deny zone Z, an **access-list permit additional-zones** entry to permit all other zones, and an **access-list permit other-access** entry to explicitly permit all networks.

## Types of Filters

You can filter the following types of AppleTalk packets:

- NBP packets
- Data packets
- Routing table updates
- GetZoneList (GZL) request and reply packets
- Zone Information Protocol (ZIP) reply packets

Table 3 shows the Cisco IOS software filters for each packet type.

*Table 3 Packet-Type-to-Filter Mapping*

| Packet Type | Filters That Can Be Applied |
|---|---|
| NBP packets | **appletalk access-group in**<br>**appletalk access-group out** |
| Data packets | **appletalk access-group in**<br>**appletalk access-group out** |
| Routing table update | **appletalk distribute-list in**<br>**appletalk distribute-list out**<br>**appletalk permit-partial-zones**<br>**appletalk zip-reply-filter** |
| ZIP reply packets | **appletalk zip-reply-filter** |
| GZL request and reply packets | **appletalk distribute-list in**<br>**appletalk distribute-list out**<br>**appletalk getzonelist-filter**<br>**appletalk permit-partial-zones** |

**Note** These types of filters are completely independent of each other, which means that if, for example, you apply a data packet filter to an interface, that filter has no effect on incoming routing table updates or GZL requests that pass through that interface. The exceptions to this rule are that outgoing routing update filters can affect GZL updates, and ZIP reply filters can affect outgoing routing updates.

## Implementation Considerations

Unlike access lists in other protocols, the order of the entries in an AppleTalk access list is not important. However, keep the following constraints in mind when defining access lists:

- You must design and type access list entries properly to ensure that entries do not overlap each other. An example of an overlap is if you were to use a **permit network** command and then use a **deny network** command. If you do use entries that overlap, the last one you used overwrites and removes the previous one from the access list. In this example, the “permit network” statement would be removed from the access list when you typed the “deny network” statement.
- Each access list always has a method for handling packets or routing updates that do not satisfy any of the access control statements in the access list.

  To explicitly specify how you want these packets or routing updates to be handled, use the **access-list other-access** global configuration command when defining access conditions for networks and cable ranges, use the **access-list additional-zones** global configuration command when defining access conditions for zones, and use the **access-list other-nbps** global configuration command when defining access conditions for NBP packets from named entities. If you use one of these commands, it does not matter where in the list you place it. The Cisco IOS software automatically places an **access-list deny other-access** command at the end of the list. It also places **access-list deny additional-zones** and **access-list deny other-nbps** commands at the end of the access list when zones and NBP access conditions are denied, respectively. (With other protocols, you must type the equivalent commands last.)

If you do not explicitly specify how to handle packets or routing updates that do not satisfy any of the access control statements in the access list, the packets or routing updates are automatically denied access and, in the case of data packets, are discarded.

## Controlling Access to AppleTalk Networks Task List

To control access to AppleTalk networks, perform the tasks in the following sections:

- Creating Access Lists (Optional)
- Creating Filters (Optional)

## Creating Access Lists

An access list defines the conditions used to filter packets sent into or out of the interface. Each access list is identified by a number. All **access-list** commands that specify the same access list number create a single access list.

A single access list can contain any number and any combination of **access-list** commands. You can include network and cable range **access-list** commands, zone **access-list** commands, and NBP named entity **access-list** commands in the same access list.

However, you can specify only one each of the commands that specify default actions to take if none of the access conditions are matched. For example, a single access list can include only one **access-list other-access** command to handle networks and cable ranges that do not match the access conditions, only one **access-list additional-zones** command to handle zones that do not match the access conditions, and only one **access-list other-nbps** command to handle NBP packets from named entities that do not match the access conditions.

You can also set priorities for the order in which outgoing packets destined for a specific network are queued, based on the access list.

**Note** For priority queueing, the Cisco IOS software applies the access list to the destination network.

AppleTalk access lists are automatically fast switched. Access list fast switching improves the performance of AppleTalk traffic when access lists are defined on an interface.

### Creating AppleTalk-Style Access Lists

To create AppleTalk-style access lists, perform the tasks in the following sections:

- Creating Zone Access Lists (Optional)
- Creating Priority Queueing Access Lists (Optional)
- Creating NBP Access Lists (Optional)

### Creating Zone Access Lists

To create access lists that define access conditions for zones (AppleTalk-style access lists), use one or more of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **zone** *zone-name* | Defines access for a zone. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **additional-zones** | Defines the default action to take for access checks that apply to zones. |

For examples of how to create access lists, see the "AppleTalk Access List Examples" and "Hiding and Sharing Resources with Access List Examples" sections at the end of this chapter.

### Creating Priority Queueing Access Lists

To assign a priority in which packets destined for a specific zone will be queued, based on the zone access list, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **priority-list** *list-number* **protocol** *protocol-name* {**high** \| **medium** \| **normal** \| **low**} **list** *access-list-number* | Defines access for a single network number. |

### Creating NBP Access Lists

To create access lists that define access conditions for NBP packets based on the NBP packet type, from particular NBP named entities, from classes of NBP named entities, or from NBP named entities within particular zones, use one or both of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **nbp** *sequence-number* {**BrRq** \| **FwdRq** \| **Lookup** \| **LkReply** \| **object** *string* \| **type** *string* \| **zone** *string*} | Defines access for an NBP packet type, NBP named entity, type of named entity, or named entities within a specific zone. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **other-nbps** | Defines the default action to take for access checks that apply to NBP named entities. |

For an example of how to create NBP packet filtering access lists, see the "Defining an Access List to Filter NBP Packets Example" section at the end of this chapter.

## Creating IP-Style Access Lists

To create access lists that define access conditions for networks and cable ranges (IP-style access lists), use one or more of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **network** *network* | Defines access for a single network number. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} *cable-range* [**broadcast-deny** \| **broadcast-permit**] | Defines access for a single cable range. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **includes** *cable-range* [**broadcast-deny** \| **broadcast-permit**] | Defines access for an extended or a nonextended network that overlaps any part of the specified range. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **within** *cable-range* [**broadcast-deny** \| **broadcast-permit**] | Defines access for an extended or a nonextended network that is included entirely within the specified range. |
| Router(config)# **access-list** *access-list-number* {**deny** \| **permit**} **other-access** | Defines the default action to take for access checks that apply to network numbers or cable ranges. |

## Creating Filters

A filter examines specific types of packets that pass through an interface and permits or denies them, based on the conditions defined in the access lists that have been applied to that interface.

To filter different types of AppleTalk packets, perform the tasks in the following sections:

- Creating NBP Packet Filters (Optional)
- Creating Data Packet Filters (Optional)
- Creating Routing Table Update Filters (Optional)
- Creating GetZoneList Filters (Optional)
- Enabling ZIP Reply Filters (Optional)
- Enabling Partial Zone Filters (Optional)

You can apply any number of filters on each interface. Each filter can use the same access list or different access lists. Filters can be applied to inbound and outbound interfaces.

Routing update filters, data packet filters, and ZIP reply filters use access lists that define conditions for networks, cable ranges, and zones. GZL filters use access lists that define conditions for zones only. NBP packet filters use access lists that define conditions for NBP named entities.

### Creating NBP Packet Filters

To create an NBP packet filter, first create an NBP access list as described in the "Creating NBP Access Lists" section earlier in this chapter and then apply an NBP filter to an interface.

To apply an NBP filter to an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| Router(config-if)# **appletalk access-group** *access-list-number* [**in** \| **out**] | Applies the data packet filter to the interface. |

**Note** Prior to Cisco IOS Release 11.2 F, all NBP access lists were applied to inbound interfaces by default. When Cisco IOS Release 11.2 F or later software is used, the default interface direction for all access lists, including NBP access lists, is outbound. In order to retain the inbound direction of access lists created with previous Cisco IOS software releases, you must specify an inbound interface for all NBP access lists by using the **appletalk access-group** command.

### Creating Data Packet Filters

A *data packet filter* checks data packets being received on an interface or sent out an interface. If the source network for the packets has access denied, these packets are discarded.

Data packet filters use access lists that define conditions for networks, cable ranges, and zones.

When you apply a data packet filter to an interface, ensure that all networks or cable ranges within a zone are governed by the same filters. For example, create a filter that works in the following way. If the router receives a packet from a network that is in a zone that contains an explicitly denied network, the router discards the packet.

To create a data packet filter, first create a network-only access list as described in the "Creating Zone Access Lists" and "Creating IP-Style Access Lists" sections earlier in this chapter and then apply a data packet filter to an interface.

To apply the data packet filter to an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)#` **`appletalk access-group`** *`access-list-number`* `[`**`in`** `|` **`out`**`]` | Applies the data packet filter to the interface. |

For an example of how to create data packet filters, see the "AppleTalk Access List Examples" section at the end of this chapter.

### Creating Routing Table Update Filters

Routing table update filters control which updates the local routing table accepts and which routes the local router advertises in its routing updates. You create distribution lists to control the filtering of routing updates.

Filters for incoming routing updates use access lists that define conditions for networks and cable ranges only. Filters for outgoing routing updates use access lists that define conditions for networks and cable ranges, and for zones.

When filtering incoming routing updates, each network number and cable range in the update is checked against the access list. If you have not applied an access list to the interface, all network numbers and cable ranges in the routing update are added to the routing table. If an access list has been applied to the interface, only network numbers and cable ranges that are not explicitly or implicitly denied are added to the routing table.

The following conditions are also applied when routing updates generated by the local router are filtered:

- The network number or cable range is not a member of a zone that is explicitly or implicitly denied.
- If partial zones are permitted, at least one network number or cable range that is a member of the zone is explicitly or implicitly permitted. If partial zones are not permitted (the default), all network numbers or cable ranges that are members of the zone are explicitly or implicitly permitted.

#### Creating Routing Table Update Filters for Incoming Updates

To create a filter for routing table updates received on an interface, create an access list as described in the "Creating IP-Style Access Lists" section earlier in this chapter and then apply a routing table update filter to an interface.

**Note** Cisco IOS software ignores zone entries. Therefore, ensure that access lists used to filter incoming routing updates do not contain any zone entries.

To apply the filter to incoming routing updates on an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)#` **`appletalk distribute-list`** *`access-list-number`* **`in`** | Applies the routing update filter. |

For an example of how to create a filter for incoming routing table updates, see the "AppleTalk Access List Examples" section at the end of this chapter.

#### Creating Routing Table Update Filters for Outgoing Updates

To create a filter for routing table updates sent out from an interface, create an access list as described in the "Creating Zone Access Lists" and "Creating IP-Style Access Lists" sections earlier in this chapter and then apply a routing table update filter to an interface.

**Note** You can use zone entries in access lists used to filter outgoing routing updates.

To apply a filter to routing updates sent out from an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)#` **`appletalk distribute-list`** *`access-list-number`* **`out`** | Applies the routing update filter. |

### Creating GetZoneList Filters

The Macintosh Chooser uses ZIP GZL requests to compile a list of zones from which the user can select services. Any router on the same network as the Macintosh can respond to these requests with a GZL reply. You can create a GZL filter to control which zones the Cisco IOS software mentions in its GZL replies. Creating this type of filter has the effect of controlling the list of zones that are displayed by the Chooser.

When defining GZL filters, you should ensure that all routers on the same network filter GZL replies identically. Otherwise, the Chooser will list different zones depending on which device responded to the request. Also, inconsistent filters can result in zones appearing and disappearing every few seconds when the user remains in the Chooser. Because of these inconsistencies, you should normally apply GZL filters only when all routers in the internetwork are Cisco routers, unless the routers from other vendors have a similar feature.

When a ZIP GZL reply is generated, only zones that satisfy the following conditions are included:

- If partial zones are permitted, at least one network number or cable range that is a member of the zone is explicitly or implicitly permitted.

- If partial zones are not permitted (the default), all network numbers or cable ranges that are members of the zone are explicitly or implicitly permitted.
- The zone is explicitly or implicitly permitted.

Replies to GZL requests also are filtered by any outgoing routing update filter that has been applied to the same interface. You must apply a GZL filter only if you want additional filtering to be applied to GZL replies. This filter is rarely needed, except to eliminate zones that do not contain user services.

Using a GZL filter is not a complete replacement for anonymous network numbers. To prevent users from seeing a zone, all routers must implement the GZL filter. If any devices on the network are from other vendors, the GZL filter will not have a consistent effect.

To create a GZL filter, create an access list as described in the "Creating Zone Access Lists" section earlier in this chapter and then apply a GZL filter to an interface.

To apply the GZL filter to an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)#` **`appletalk getzonelist-filter`** *`access-list-number`* | Applies the GZL filter. |

For an example of how to create a GZL filters, see the "GZL and ZIP Reply Filter Examples" section at the end of this chapter.

### Enabling ZIP Reply Filters

ZIP reply filters limit the visibility of zones from routers in unprivileged regions throughout the internetwork. These filters filter the zone list for each network provided by a router to neighboring devices to remove restricted zones.

ZIP reply filters apply to downstream routers, not to end stations on networks attached to the local router. With ZIP reply filters, when downstream routers request the names of zones in a network, the local router replies with the names of visible zones only. It does not reply with the names of zones that have been hidden with a ZIP reply filter. To filter zones from end stations, use GZL filters.

ZIP reply filters determine which networks and cable ranges the Cisco IOS software sends out in routing updates. Before sending out routing updates, the software excludes the networks and cable ranges whose zones have been completely denied access by ZIP reply filters. Excluding this information ensures that routers receiving these routing updates do not send unnecessary ZIP requests.

To create a ZIP reply filter, create an access list as described in the "Creating Zone Access Lists" section earlier in this chapter and then apply a ZIP reply filter to an interface.

To apply the ZIP reply filter to an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)#` **`appletalk zip-reply-filter`** *`access-list-number`* | Applies the ZIP reply filter. |

For an example of how to create GZL and ZIP reply filters, see the "GZL and ZIP Reply Filter Examples" section at the end of this chapter.

### Enabling Partial Zone Filters

If access to any network in a zone is denied, access to that zone is also denied by default. However, if you enable partial zones, access to other networks in that zone is no longer denied.

The permitting of partial zones provides IP-style access control. If enabled, the access control list behavior associated with prior software releases is restored. In addition, NBP cannot ensure consistency and uniqueness of name bindings.

If you permit partial zones, AppleTalk cannot maintain consistency for the nodes in the affected zones, and the results are undefined. With this option enabled, an inconsistency is created for the zone, and several assumptions made by some AppleTalk protocols are no longer valid.

To enable partial zone filters, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk permit-partial-zones`** | Permits access to networks in a zone in which access to another network in that zone is denied. |

Permitting partial zones affects the outgoing routing update and GZL filters.

# Configuring the Name Display Facility

The AppleTalk NBP associates AppleTalk network entity names (that is, AppleTalk network-addressable services) with network addresses. NBP allows you to specify descriptive or symbolic names for entities instead of their numerical addresses. When you specify the name of an AppleTalk device, NBP translates the entity name of the device into the network address of the device. The name binding process includes name registration, name confirmation, name deletion, and name lookup.

Node addresses can change frequently because AppleTalk uses dynamic addresses. Therefore, NBP associates numerical node addresses with aliases that continue to reference the correct addresses if the addresses change. These node addresses do not change very frequently because each device keeps track of the last node number it was assigned. Typically, node numbers change only if a device is shut down for an extended period of time, or if the device is moved to another network segment.

To control the name display facility, use one or both of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk lookup-type`** *`service-type`* | Specifies which service types are retained in the name cache. |
| `Router(config)#` **`appletalk name-lookup-interval`** *`seconds`* | Sets the interval between service pollings by the router on its AppleTalk interfaces. |

# Setting Up Special Configurations

To set up special configurations, perform the tasks in the following sections, based on desired service implementations:

- Configuring Free-Trade Zones (Optional)
- Configuring SNMP over DDP in AppleTalk Networks (Optional)
- Configuring AppleTalk Tunneling (Optional)
- Configuring AppleTalk MacIP (Optional)

- Configuring IPTalk (Optional)

## Configuring Free-Trade Zones

A free-trade zone is a part of an AppleTalk internetwork that is accessible by two other parts of the internetwork, neither of which can access the other. You might want to create a free-trade zone to allow the exchange of information between two organizations that otherwise want to keep their internetworks isolated from each other, or that do not have physical connectivity with one another.

To establish a free-trade zone, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk free-trade-zone` | Establishes a free-trade zone. |

For an example of how to configure a free-trade zone, see the "Hiding and Sharing Resources with Access List Examples" section and the "Establishing a Free-Trade Zone Example" section at the end of this chapter.

## Configuring SNMP over DDP in AppleTalk Networks

The Simple Network Management Protocol (SNMP) normally uses the IP connectionless datagram service, the User Datagram Protocol (UDP), to monitor network entities. The Cisco IOS software lets you run SNMP using Datagram Delivery Protocol (DDP), the AppleTalk datagram service. Use DDP if you have SNMP consoles running on a Macintosh.

You must configure AppleTalk routing globally and on an interface basis before you configure SNMP for the router; therefore, you need to disable SNMP as shown in the following command table.

To configure SNMP in AppleTalk networks, use the following commands beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | Router(config)# **no snmp server** | Disables SNMP. |
| **Step 2** | Router(config)# **appletalk routing** | Enables AppleTalk routing. |
| **Step 3** | Router(config)# **appletalk event-logging** | Enables AppleTalk event logging. |
| **Step 4** | Router(config)# **interface** *type number* | Enters interface configuration mode. |
| **Step 5** | Router(config-if)# **ip address** *ip-address mask* | Enables IP routing on the interface. |
| **Step 6** | Router(config-if)# **appletalk cable-range** *cable-range* [*network*.*node*] | Enables AppleTalk routing on the interface. |
| **Step 7** | Router(config-if)# **appletalk zone** *zone-name* | Sets a zone name for the AppleTalk network. |
| **Step 8** | Router(config-if)# **snmp-server community** *string* [**RO**] [**RW**] [*number*] | Enables SNMP server operations. |

For an example of how to configure SNMP, see the "SNMP Example" section at the end of this chapter.

For information about configuring SNMP, refer to the "Monitoring the Router and Network" chapter in the *Cisco IOS Configuration Fundamentals Configuration Guide*.

## Configuring AppleTalk Tunneling

Tunneling provides a means for encapsulating packets inside a routable protocol via virtual interfaces. Encapsulation takes packets or frames from one network system and places them inside frames from another network system. There are three ways to configure AppleTalk tunneling so that you can connect remote AppleTalk networks across a foreign protocol backbone such as the Internet or IP:

- Configuring AURP
- Configuring GRE
- Configuring Cayman Tunneling

The method of tunneling is chosen based on the end destination and your encapsulation type.

Multiple tunnels originating from the router are supported. Logically, tunnels are point-to-point links and therefore require that you configure a separate tunnel for each link.

If you are experiencing traffic congestion due to RTMP overhead, you can resolve this problem by using one of two AppleTalk tunneling methods—AppleTalk Update-Based Routing Protocol (AURP) or GRE tunneling. The AppleTalk packets will be tunneled through a foreign protocol, such as IP. Tunneling encapsulates an AppleTalk packet inside the foreign protocol packet, which is then sent across the backbone to a destination router. The destination router then de-encapsulates the AppleTalk packet and, if necessary, routes the packet to a normal AppleTalk network. The encapsulated packet benefits from any features normally enjoyed by IP packets, including default routes and load balancing.

### Configuring AURP

The first and most often recommended AppleTalk tunneling method is to enable AppleTalk Update-Based Routing Protocol (AURP). When two AppleTalk networks are connected with a non-AppleTalk backbone such as IP, the relatively high bandwidth consumed by the broadcasting of RTMP data packets may impact the network performance of the backbone. Using AURP will lower the routing protocol overhead across a WAN or backbone because it changes the encapsulation method as well as the routing algorithm to something more like link state routing.

**Note** Bandwidth is usually more constrained in a WAN than on a backbone.

AURP is a standard Apple Computer routing protocol that provides enhancements to the AppleTalk routing protocols that are compatible with AppleTalk Phase 2. The primary function of AURP is to connect two or more noncontiguous AppleTalk internetworks that are separated by a non-AppleTalk network (such as IP). In these configurations, you would want to use AURP instead of RTMP, because AURP sends fewer routing packets than RTMP.

You configure AURP on a tunnel interface. Tunneling encapsulates an AppleTalk packet inside an IP packet, which is sent across the backbone to a destination router. The destination device then extracts the AppleTalk packet and, if necessary, routes it to an AppleTalk network. The encapsulated packet benefits from any features normally applied to IP packets, including fragmentation, default routes, and load balancing.

After you configure an AppleTalk domain for AppleTalk interenterprise features, you can apply the features to a tunnel interface configured for AURP by assigning the domain number to the interface.

Because route redistribution is disabled by default, you need to enable it by using the **appletalk route-redistribution** command.

To configure AURP, use the following commands beginning in global configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config)# appletalk route-redistribution` | Enables route redistribution. |
| **Step 2** | `Router(config)# interface type number` | Configures an interface to be used by the tunnel. |
| **Step 3** | `Router(config-if)# ip address ip-address mask` | Configures an IP address. |
| **Step 4** | `Router(config-if)# interface tunnel number` | Configures tunnel interface. |
| **Step 5** | `Router(config-if)# appletalk protocol aurp` | Creates an AURP routing process. |
| **Step 6** | `Router(config-if)# tunnel source {ip-address | type number}` | Specifies the interface out of which the encapsulated packets will be sent. |
| **Step 7** | `Router(config-if)# tunnel destination {hostname | ip-address}` | Specifies the IP address of the router at the far end of the tunnel. |
| **Step 8** | `Router(config-if)# tunnel mode aurp` | Enables AURP tunneling. |

You can configure AURP on a tunnel interface to inherit AppleTalk interenterprise routing remapping, hop count reduction, and loop detection characteristics configured for a specific AppleTalk domain. To do so, these features must first be configured for the AppleTalk domain using the commands described in the tasks “Enabling AppleTalk Interenterprise Routing,” “Remapping Network Numbers,” and “Controlling Hop Count” within the section “Configuring AppleTalk Interenterprise Routing” later in this chapter.

To configure AURP for AppleTalk interenterprise routing features, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# interface tunnel number` | Specifies the tunnel interface. |
| **Step 2** | `Router(config-if)# appletalk protocol aurp` | Creates an AURP routing process. |
| **Step 3** | `Router(config-if)# tunnel mode aurp` | Enables AURP tunneling. |
| **Step 4** | `Router(config-if)# tunnel source {ip-address | type number}` | Specifies the interface out of which the encapsulated packets will be sent. |
| **Step 5** | `Router(config-if)# tunnel destination {hostname | ip-address}` | Specifies the IP address of the router at the far end of the tunnel. |
| **Step 6** | `Router(config-if)# appletalk domain-group domain-number` | Assigns the number of the predefined AppleTalk domain to which the AppleTalk interenterprise features are configured to the tunnel interface configured for AURP. |

For an example of how to configure AURP on a tunnel interface to inherit AppleTalk interenterprise routing features for a specific AppleTalk domain, see the “AppleTalk Interenterprise Routing over AURP Example” section at the end of this chapter.

By default, AURP sends routing updates every 30 seconds. To modify this interval, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk aurp update-interval seconds` | Sets the minimum interval between AURP routing updates. |

To set the AURP last-heard-from timer value, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk aurp tickle-time seconds` | Sets the AURP last-heard-from timer value. |

### Configuring GRE

The second AppleTalk tunneling method, a proprietary tunnel protocol known as generic routing encapsulation (GRE), is recommended when you want to use tunneling to connect one Cisco router to another. When you use GRE tunneling, you must have Cisco routers at both ends of the tunnel connection. You can also reduce RTMP overhead by using GRE tunneling. Since you do not need to run RTMP through GRE tunnels, you can significantly improve the network traffic.

To configure a GRE tunnel, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# interface tunnel number` | Configures a tunnel interface. |
| **Step 2** | `Router(config-if)# tunnel source {ip-address \| type number}` | Specifies the interface out of which the encapsulated packets will be sent. |
| **Step 3** | `Router(config-if)# tunnel destination {hostname \| ip-address}` | Specifies the IP address of the router at the far end of the tunnel. |
| **Step 4** | `Router(config-if)# tunnel mode gre ip` | Enables GRE tunneling. |

### Configuring Cayman Tunneling

The third AppleTalk tunneling method, Cayman tunneling, enables routers to interoperate with Cayman GatorBoxes. Cayman tunneling is used to connect remote AppleTalk networks across a foreign protocol backbone, such as the Internet or a backbone that is IP-only, for administrative or security reasons. You can tunnel AppleTalk by using Cayman tunneling as designed by Cayman Systems.

When you use Cayman tunneling, you can have Cisco routers at either end of the tunnel, or you can have a GatorBox at one end and a Cisco router at the other end.

To configure a Cayman tunnel, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# interface tunnel number` | Configures a tunnel interface. |
| **Step 2** | `Router(config-if)# tunnel source {ip-address \| type number}` | Specifies the interface out of which the encapsulated packets will be sent. |
| **Step 3** | `Router(config-if)# tunnel destination {hostname \| ip-address}` | Specifies the IP address of the router at the far end of the tunnel. |
| **Step 4** | `Router(config-if)# tunnel mode cayman` | Enables Cayman tunneling. |

⚠ **Caution** Do not configure a Cayman tunnel with an AppleTalk network address.

## Configuring AppleTalk MacIP

Cisco IOS software implements MacIP, which is a protocol that allows routing of IP datagrams to IP clients using the DDP for low-level encapsulation.

### The Cisco Implementation of AppleTalk MacIP

Cisco IOS software implements the MacIP address management and routing services described in the draft Internet RFC, *A Standard for the Transmission of Internet Packets over AppleTalk Networks*. Our implementation of MacIP conforms to the September 1991 draft RFC with the following exceptions:

- The software does not fragment IP datagrams that exceed the DDP maximum transmission unit (MTU) and that are bound for DDP clients of MacIP.
- The software does not route to DDP clients outside of configured MacIP client ranges.

### When to Use AppleTalk MacIP

Some situations require the use of MacIP. For example, if some of your Macintosh users use AppleTalk Remote Access (ARA) or are connected to the network using LocalTalk or PhoneNet cabling systems, then MacIP is required to provide access to IP network servers for those users.

MacIP services also can be useful when you are managing IP address allocations for a large, dynamic Macintosh population.

### Advantages of Using MacIP

The following are advantages to using MacIP when you are managing IP address allocations for a large, dynamic Macintosh population:

- Macintosh TCP/IP drivers can be configured in a completely standard way, regardless of the location of the Macintosh. Essentially, the dynamic properties of AppleTalk address management become available for IP address allocation.
- You can modify all global parameters, such as IP subnet masks, Domain Name System (DNS) services, and default routers. Macintosh IP users receive the updates by restarting their local TCP/IP drivers.
- The network administrator can monitor MacIP address allocations and packet statistics remotely by using the Telnet application to attach to the console, allowing central administration of IP allocations in remote locations. For Internet sites, it allows remote technical assistance.

### Implementation Considerations

Consider the following items when implementing MacIP on Cisco routers:

- Each packet from a Macintosh client destined for an IP host or vice versa *must* pass through the router if the client is using the device as a MacIP server. The router is not always a necessary hop, so passing through the router increases traffic through the device. There is also a slight increase in CPU use that is directly proportional to the number of packets delivered to and from active MacIP clients.
- Memory usage increases in direct proportion to the total number of active MacIP clients (about 80 bytes per client).

Also, when you configure MacIP on the Cisco IOS software, you must configure AppleTalk as follows:

- AppleTalk routing must be enabled on at least one interface.

- IP routing must be enabled on at least one interface.
- The MacIP zone name you configure must be associated with a configured or *seeded* zone name.
- The MacIP server must reside in the AppleTalk zone.
- Any IP address specified in configuring a MacIP server using an **appletalk macip** command must be associated to a specific IP interface on the router. Because the Cisco IOS software is acting as a proxy for MacIP clients, you must use an IP address to which Address Resolution Protocol (ARP) can respond.
- If you are using MacIP to allow Macintoshes to communicate with IP hosts on the same LAN segment (that is, the Macintoshes are on the router interface on which MacIP is configured) and the IP hosts have extended IP access lists, these access lists should include entries to permit IP traffic destined for these IP hosts from the MacIP addresses. If these entries are not present, packets destined for IP hosts on the local segment will be blocked (that is, they will not be forwarded).

When setting up MacIP routing, keep the following address range issues in mind:

- Static and dynamic resource statements are cumulative, and you can specify as many as necessary. However, if possible, you should specify a single all-inclusive range rather than several adjacent ranges. For example, specifying the range 172.31.121.1 to 172.31.121.10 is preferable to specifying the ranges 172.31.121.1 to 172.31.121.5 and 172.31.121.6 to 172.31.121.10.
- Overlapping resource ranges (for example, 172.31.121.1 to 172.31.121.5 and 172.31.121.5 to 172.31.121.10) are *not* allowed. If it is necessary to change a range in a running server, use the negative form of the resource address assignment command (such as **no appletalk macip dynamic** *ip-address ip-address* **zone** *server-zone*) to delete the original range, followed by the corrected range statement.
- You can add IP address allocations to a running server at any time as long as the new address range does not overlap with one of the current ranges.

## Configuring AppleTalk MacIP Task List

To configure MacIP, perform the tasks in the following sections:

- Establishing a MacIP Server for a Zone (Required)
- Allocating IP Addresses for Macintosh Users (Required)

### Establishing a MacIP Server for a Zone

To establish a MacIP server for a specific zone, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk macip server`** *`ip-address`* **`zone`** *`server-zone`* | Establishes a MacIP server for a zone. |

**Note** Note that the MacIP server must reside in the default AppleTalk zone.

You can configure multiple MacIP servers for a router, but you can assign only one MacIP server to a zone, and you can assign only one IP interface to a MacIP server. In general, you must be able to establish an alias between the IP address you assign with the **appletalk macip server** global configuration command and an existing IP interface. For implementation simplicity, the address you specify in this command should match an existing IP interface address.

A server is not registered by NBP until at least one MacIP resource is configured.

## Allocating IP Addresses for Macintosh Users

You allocate IP addresses for Macintosh users by specifying at least one *dynamic* or *static* resource address assignment command for each MacIP server.

### Allocating IP Addresses Using Dynamic Addresses

*Dynamic clients* are those that accept any IP address assignment within the dynamic range specified. *Dynamic addresses* are for users that do not require a fixed address, but can be assigned addresses from a pool.

To allocate IP addresses for Macintosh users if you are using dynamic addresses, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk macip dynamic`** *`ip-address`* `[`*`ip-address`*`]` **`zone`** *`server-zone`* | Allocates an IP address to a MacIP client. |

For an example of configuring MacIP with dynamic addresses, see the "AppleTalk Interenterprise Routing over AURP Example" section at the end of this chapter.

### Allocating IP Addresses Using Static Addresses

*Static addresses* are for users that require fixed addresses for IP DNS services and for administrators that do not want addresses to change so they always know the IP addresses of the devices on their network.

To allocate IP addresses for Macintosh users if you are using static addresses, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk macip static`** *`ip-address`* `[`*`ip-address`*`]` **`zone`** *`server-zone`* | Allocates an IP address to be used by a MacIP client that has reserved a static IP address. |

For an example of configuring MacIP with static addresses, see the "MacIP Examples" section at the end of this chapter.

In general, it is recommended that you do not use fragmented address ranges in configuring ranges for MacIP. However, if fragmented address ranges are unavoidable, use the **appletalk macip dynamic** command to specify as many addresses or ranges as required, and use the **appletalk macip static** command to assign a specific address or address range.

## Configuring IPTalk

IPTalk is a protocol for encapsulating AppleTalk packets in IP datagrams. IPTalk is used to route AppleTalk packets across non-AppleTalk backbones and to communicate with applications on hosts that cannot otherwise communicate via AppleTalk, such as the Columbia AppleTalk Package (CAP). IPTalk also allows serial connections to use IPTalk Serial Line Internet Protocol (SLIP) drivers.

If your system is a Sun or Digital Equipment Corporation ULTRIX system, it may be possible to run CAP directly in a mode that supports EtherTalk. In this case, your system would look like any other AppleTalk node and does not need any special IPTalk support. However, other UNIX systems for which EtherTalk support is not available in CAP must run CAP in a mode that depends upon IPTalk.

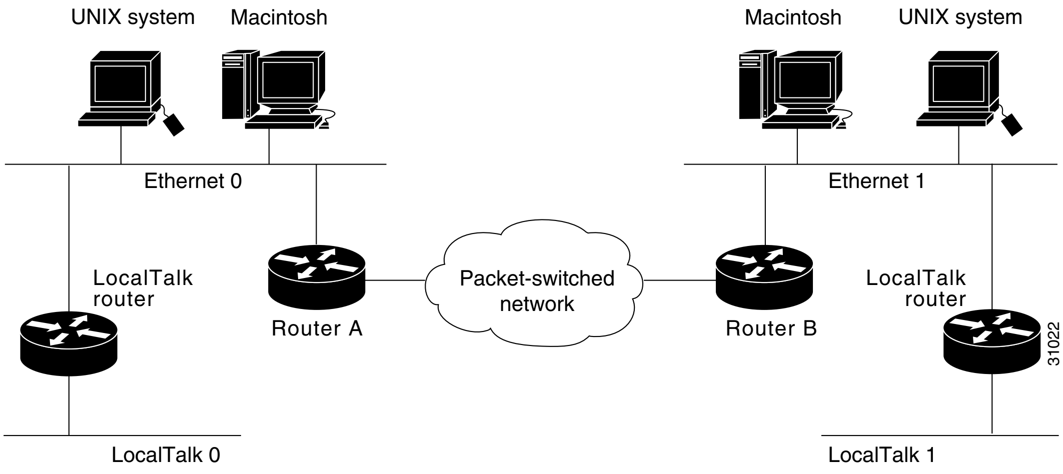
For installation instructions for CAP, refer to Kinetics IP (KIP) gateways and the file *atalkatab*. If you use Cisco IPTalk support, it is not necessary (nor is it desirable) to use *atalkatab*. Cisco IPTalk support assumes that you want to use the standard AppleTalk routing protocols to perform all wide-area AppleTalk routing. KIP and *atalkatab* are based on an alternative routing strategy in which AppleTalk packets are sent using IP routing. It is possible to use both strategies at the same time; however, the interaction between the two routing techniques is not well defined.

If your network has routers from other vendors that support *atalkatab*, you should disable *atalkatab* support on them to avoid mixing the routing strategies. The installation instructions provided with some of these products encourage you to use *atalkatab* for complex networks. However, with Cisco routers it is not necessary, because our implementation of IPTalk integrates IPTalk into the standard AppleTalk network routing.

The network diagram in Figure 1 illustrates how you should set up IPTalk. In this configuration, you enable both standard AppleTalk (EtherTalk) and IPTalk on the Ethernet networks on Router A and Router B. These routers then use EtherTalk to communicate with the LocalTalk routers and Macintosh computers, and IPTalk to communicate with the UNIX systems. On the LocalTalk routers, you also should enable both EtherTalk and IPTalk, making sure you configure IPTalk with *atalkatab* disabled. These routers then use IPTalk to communicate with the UNIX systems adjacent to them and EtherTalk to communicate with the remainder of the AppleTalk network. This configuration strategy minimizes the number of hops between routers. If you did not enable IPTalk on the LocalTalk routers, systems on the LocalTalk router that wanted to communicate with the adjacent UNIX system would need to go through Router A or Router B, creating an unnecessary extra hop.

**Note** In the configuration shown in Figure 1, all traffic between systems on the left and right sides of the packet-switched network transit via Router A and Router B using AppleTalk routing. If you were to enable *atalkatab* support on the LocalTalk routers, a hidden path would be established between Router A and Router B, unknown to the standard AppleTalk routing protocols. In a large network, this hidden path could result in traffic taking inexplicable routes.

*Figure 1 IPTalk Configuration Example*



To configure IPTalk on an interface, perform the following tasks:

- Configuring IP Encapsulation of AppleTalk Packets (Required)
- Specifying the UDP Port Ranges (Required)

## Configuring IP Encapsulation of AppleTalk Packets

To allow AppleTalk to communicate with UNIX hosts running older versions of CAP that do not support native AppleTalk EtherTalk encapsulations, you must configure IP encapsulation of AppleTalk packets. (Typically, Apple Macintosh users would communicate with these servers by routing their connections through a Kinetics FastPath router running KIP software.) Newer versions of CAP provide native AppleTalk EtherTalk encapsulations, so the IPTalk encapsulation is no longer required. The Cisco implementation of IPTalk assumes that AppleTalk is already being routed on the backbone, because there is currently no LocalTalk hardware interface for our routers.

You configure IPTalk on a tunnel interface. Tunneling encapsulates an AppleTalk packet inside an IP packet, which is sent across the backbone to a destination router. The destination device then extracts the AppleTalk packet and, if necessary, routes it to an AppleTalk network. The encapsulated packet benefits from any features normally applied to IP packets, including fragmentation, default routes, and load balancing.

The Cisco implementation of IPTalk does not support manually configured AppleTalk-to-IP-address mapping. The address mapping provided is the same as the Kinetics IPTalk implementation when AppleTalk-to-IP-address mapping is not enabled. This address mapping works as follows:

1. The IP subnet mask used on the router tunnel source interface on which IPTalk is enabled is inverted (ones complement).
2. The result is then masked against 255 (0xFF hexadecimal).
3. The result of this is then masked against the low-order 8 bits of the IP address to give the AppleTalk node number.

The following example shows how to configure address mapping:

```
interface Ethernet0
ip address 172.16.1.118 255.255.255.0
appletalk address 20.129
appletalk zone Native AppleTalk
interface Tunnel0
tunnel source Ethernet0
tunnel mode iptalk
appletalk iptalk 30 UDPZone
```

First, the IP subnet mask of 255.255.255.0 is inverted to 0.0.0.255. This value is then masked with 255 to give 255. Next, 255 is masked with the low-order 8 bits of the interface IP address (118) to yield an AppleTalk node number of 118, which means that the AppleTalk address of the Ethernet interface 0 seen in the UDPZone zone is 30.118.

**Note** If the host field of an IP subnet mask for an interface is longer than 8 bits, it will be possible to obtain conflicting AppleTalk node numbers. For instance, if the subnet mask for the Ethernet interface 0 above is 255.255.240.0, the host field is 12 bits wide.

To configure IP encapsulation of AppleTalk packets, use the following commands in interface configuration mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | Router(config-if)# **interface** *type number* | Configures an interface to be used by the tunnel. |
| **Step 2** | Router(config-if)# **ip address** *ip-address mask* | Configures an IP address. |
| **Step 3** | Router(config-if)# **interface tunnel** *number* | Configures tunnel interface. |
| **Step 4** | Router(config-if)# **tunnel source** {*ip-address* \| *type number*} | Specifies the interface out of which the encapsulated packets will be sent. |
| **Step 5** | Router(config-if)# **tunnel mode iptalk** | Enables IPTalk tunneling. |

For an example of configuring IPTalk, see the "IPTalk Example" section at the end of this chapter.

### Specifying the UDP Port Ranges

Implementations of IPTalk prior to April 1988 mapped well-known DDP socket numbers to privileged UDP ports starting at port number 768. In April 1988, the Network Information Center (NIC) assigned a range of UDP ports for the defined DDP well-known sockets starting at UDP port number 200 and assigned these ports the names at-nbp, at-rtmp, at-echo, and at-zis. Release 6 and later of the CAP program dynamically decides which port mapping to use. If there are no AppleTalk service entries in the /etc/services file of the UNIX system, CAP uses the older mapping starting at UDP port number 768.

The default UDP port mapping supported by our implementation of IPTalk is 768. If there are AppleTalk service entries in the /etc/services file of the UNIX system, you should specify the beginning of the UDP port mapping range.

To specify the UDP port number that is the beginning of the range of UDP ports used in mapping AppleTalk well-known DDP socket numbers to UDP ports, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk iptalk-baseport** | Specifies the starting UDP port number. |

For an example of configuring IPTalk, see the "IPTalk Example" section at the end of this chapter.

## Configuring AppleTalk Control Protocol for PPP

You can configure an asynchronous interface (including the auxiliary port on some Cisco routers) to use AppleTalk Control Protocol (ATCP) so that users can access AppleTalk zones by dialing into the router via PPP to this interface. Asynchronous interfaces are configured with ATCP through a negotiation protocol, as defined in RFC 1378. Users accessing the network with ATCP can run AppleTalk and IP natively on a remote Macintosh, access any available AppleTalk zones from the Chooser, use networked peripherals, and share files with other Macintosh users.

You create an internal network with the **appletalk internal-network** command. This network is a virtual network and exists only for accessing an AppleTalk internetwork through the server.

To create a new AppleTalk zone, enter the **appletalk virtual-net** command and use a new zone name; this network number is then the only one associated with this zone. To add network numbers to an existing AppleTalk zone, use the existing zone name in the command; the network number is then added to the existing zone.

Routing is not supported on these interfaces.

To enable ATCP for PPP, use the following commands in interface configuration (asynchronous) mode:

| | Command | Purpose |
|---|---|---|
| **Step 1** | `Router(config-if)# interface async number` | Specifies an asynchronous interface. |
| **Step 2** | `Router(config-if)# appletalk virtual-net network-number zone-name` | Creates an internal network on the server. |
| **Step 3** | `Router(config-if)# encapsulation ppp` | Enables PPP encapsulation on the interface. |
| **Step 4** | `Router(config-if)# appletalk client-mode` | Enables client-mode on the interface. |

For an example of configuring ATCP, see the "AppleTalk Control Protocol Example" section at the end of this chapter.

## Tuning AppleTalk Network Performance

To tune AppleTalk network performance, perform one or more of the tasks described in the following sections:

- Controlling Routing Updates (Optional)
- Assigning Proxy Network Numbers (Optional)
- Enabling Round-Robin Load Sharing (Optional)
- Disabling Checksum Generation and Verification (Optional)
- Controlling the AppleTalk ARP Table (Optional)
- Controlling the Delay Between ZIP Queries (Optional)
- Logging Significant Network Events (Optional)
- Disabling Fast Switching (Optional)

## Controlling Routing Updates

RTMP establishes and maintains the AppleTalk routing table. To control packet routing and control routing updates, perform the tasks in the following sections:

- Disabling the Processing of Routed RTMP Packets (Optional)
- Enabling RTMP Stub Mode (Optional)
- Disabling the Transmission of Routing Updates (Optional)
- Preventing the Advertisement of Routes to Networks with No Associated Zones (Optional)
- Setting Routing Table Update Timers (Optional)
- Setting the Routing Update Interval Timer (Optional)

### Disabling the Processing of Routed RTMP Packets

By default, the Cisco IOS software performs strict RTMP checking, which discards any RTMP packets sent by routers not directly connected to the local device (that is, sent by devices that are not neighbors). In this case, the local router does not accept any routed RTMP packets whose source is a remote network.

In almost all situations, you should leave RTMP checking enabled.

To disable RTMP checking and enable the processing of routed RTMP packets, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# no appletalk strict-rtmp-checking` | Disables strict checking of RTMP updates. |

### Enabling RTMP Stub Mode

You can enable AppleTalk RTMP stub mode. This mode allows routers running Enhanced IGRP and RTMP to reduce the amount of CPU time that RTMP modules use. In this mode, RTMP modules send and receive only "stub" RTMP packets.

A stub packet is only the first tuple of an RTMP packet. The first tuple indicates the network number range assigned to that network. End nodes use stub packets to determine if their node number is in the correct network range.

To enable AppleTalk RTMP stub mode, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk rtmp-stub` | Enables RTMP stub mode. |

### Disabling the Transmission of Routing Updates

By default, routers receive routing updates from their neighboring devices and periodically send routing updates to their neighbors. You can configure the Cisco IOS software so that it only receives routing updates, but does not send any updates. You might want to use this type of configuration to keep a particular router that is unreliable from sending routing updates to its neighbors.

To disable the transmission of routing updates, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| Router(config-if)# **no appletalk send-rtmps** | Disables the transmission of routing updates on an interface. |

### Preventing the Advertisement of Routes to Networks with No Associated Zones

NBP uses ZIP to determine which networks belong to which zones. The Cisco IOS software uses ZIP to maintain a table of the AppleTalk internetwork that maps network numbers to zone names.

By default, the software does not advertise routes to networks that have no associated zones and therefore prevents the occurrence of ZIP protocol storms, which can arise when corrupt routes are propagated and routers broadcast ZIP requests to determine the network-zone associations. By not advertising routes to networks that do not have associated zones, you limit any ZIP protocol storms to a single network, rather than allowing them to spread to the entire internetwork.

To allow the advertisement of routes to networks that have no associated zones, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **no appletalk require-route-zones** | Allows the advertisement of routes to networks that have no associated zones. |

The *user* zone lists can be configured to vary from interface to interface. However, this practice is discouraged because AppleTalk users expect to have the same user zone lists at any end node in the internetwork. This kind of filtering does not prevent explicit access via programmatic methods, but should be considered a user optimization whereby unused zones are suppressed. Use other forms of AppleTalk access control lists to actually *secure* a zone or network.

### Setting Routing Table Update Timers

Cisco IOS software sends routing table updates at regular intervals. In rare instances, you might want to change this interval, such as when a router is busy and cannot send routing updates every 10 seconds, or when slower devices are incapable of processing received routing updates in a large network. If you do change the routing update interval, you must do so for *all* devices on the network.

**Caution** Modifying the routing timers can degrade or destroy AppleTalk network connectivity. Many other AppleTalk router vendors provide no facility for modifying their routing timers, so adjusting Cisco AppleTalk timers such that routing updates do not arrive at these other routers within the normal interval might result in loss of information about the network or loss of connectivity.

To change the routing table update timers, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk timers** *update-interval valid-interval invalid-interval* | Changes the routing update timers. |

### Setting the Routing Update Interval Timer

The interval between subsequent routing updates is randomized to reduce the probability of synchronization with the routing updates from other routers on the same link. This randomization is achieved by maintaining a separate transmission interval timer for each advertising interface.

To set the interval timer on a router between subsequent routing updates, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk rtmp jitter`** *`percent`* | Sets the interval timer between subsequent routing updates. |

## Assigning Proxy Network Numbers

It is possible to have an AppleTalk internetwork in which some routers support only nonextended AppleTalk and others support only extended AppleTalk. You can enable interoperability between these two types of AppleTalk networks by assigning a proxy network number for each zone in which there is a device that supports only nonextended AppleTalk.

To assign proxy network numbers, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk proxy-nbp`** *`network-number zone-name`* | Assigns a proxy network number for each zone in which there is a device that supports only nonextended AppleTalk. |

For an example of how to configure proxy network numbers, see the "Proxy Network Number Example" section at the end of this chapter.

**Caution** Do not also assign the proxy network number to a router or to a physical network.

You must assign one proxy network number for each zone. You can optionally define additional proxies with different network numbers to provide redundancy. Each proxy network number generates one or more packets for each forward request it receives, but discards all other packets sent to it. Thus, defining redundant proxy network numbers increases the NBP traffic linearly.

## Enabling Round-Robin Load Sharing

In order to increase throughput in the network, a router can use multiple equal-cost paths to reach a destination. By default, the router picks one best path and sends all traffic using this path. You can configure the router to remember two or more paths that have equal costs, and to balance the traffic load across all of the available paths. (Note that when paths have differing costs, the Cisco IOS software chooses lower-cost routes in preference to higher-cost routes.)

The software then distributes output on a packet-by-packet basis in round-robin fashion. That is, the first packet is sent along the first path, the second packet along the second path, and so on. When the final path is reached, the next packet is sent to the first path, the next to the second path, and so on. This round-robin scheme is used regardless of whether fast switching is enabled.

Limiting the number of equal-cost paths can save memory on routers with limited memory or with very large configurations. Additionally, in networks with a large number of multiple paths and systems with limited ability to cache out-of-sequence packets, performance might suffer when traffic is split between many paths.

To set the maximum number of paths, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk maximum-paths paths` | Sets the maximum number of equal-cost paths to a destination. |

## Disabling Checksum Generation and Verification

By default, the Cisco IOS software generates and verifies checksums for all AppleTalk packets (except routed packets). You might want to disable checksum generation and verification if you have older devices (such as LaserWriter printers) that cannot receive packets with checksums.

To disable checksum generation and verification, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# no appletalk checksum` | Disables the generation and verification of checksums for all AppleTalk packets. |

## Controlling the AppleTalk ARP Table

To control the AppleTalk ARP table, you can use the following tasks:

- Set the timeout for ARP table entries
- Specify the time interval between the retransmission of ARP packets
- Specify the number of ARP retransmissions
- Disable the gleaning of ARP information from incoming packets

By default, entries in the AppleTalk ARP table are removed from the table if no update has been received in the last 4 hours. To change the ARP timeout interval, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk arp-timeout interval` | Sets the timeout for ARP table entries. |

AppleTalk ARP associates AppleTalk network addresses with media (data link) addresses. When AppleTalk must send a packet to another network node, the protocol address is passed to AppleTalk ARP, which undertakes a series of address negotiations to associate the protocol address with the media address.

If your AppleTalk network has devices that respond slowly (such as printers and overloaded file servers), you can lengthen the interval between AppleTalk ARP packets in order to allow the responses from these devices to be received. To lengthen the interval between AppleTalk ARP packets, use one or both of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk arp** [**probe** \| **request**] **interval** *interval* | Specifies the time interval between retransmission of ARP packets. |
| Router(config)# **appletalk arp** [**probe** \| **request**] **retransmit-count** *number* | Specifies the number of retransmissions that will occur before abandoning address negotiations and using the selected address. |

The Cisco IOS software automatically derives ARP table entries from incoming packets. This process is referred to as *gleaning*. Gleaning speeds up the process of populating the ARP table. To disable the gleaning of ARP table entries, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| Router(config-if)# **no appletalk glean-packets** | Disables the gleaning of ARP information from incoming packets. |

## Controlling the Delay Between ZIP Queries

By default, the Cisco IOS software sends ZIP queries every 10 seconds and uses the information received to update its zone table. To change the ZIP query interval, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk zip-query-interval** *interval* | Sets the ZIP query interval. |

## Logging Significant Network Events

You can log information about significant network events performed on the router, including routing changes, zone creation, port status, and address. To log information about significant network events, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| Router(config)# **appletalk event-logging** | Logs significant events. |

## Disabling Fast Switching

Fast switching allows higher throughput by switching a packet using a cache created by previous packets. Fast switching is enabled by default on all interfaces that support fast switching.

Packet transfer performance is generally better when fast switching is enabled. However, you may want to disable fast switching in order to save memory space on interface cards and to help avoid congestion when high-bandwidth interfaces are writing large amounts of information to low-bandwidth interfaces.

To disable AppleTalk fast switching on an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# no appletalk route-cache` | Disables AppleTalk fast switching. |

# Configuring AppleTalk Interenterprise Routing

AppleTalk interenterprise routing provides support for AppleTalk internets, or *domains*. AppleTalk interenterprise routing allows two or more AppleTalk domains to be connected through a domain router (which can also be a Cisco access server). AppleTalk interenterprise routing allows the resolution of conflicting AppleTalk network numbers or cable ranges from different domains and hop-count reduction between domains.

## Understanding AppleTalk Domains

An AppleTalk domain is a group of AppleTalk networks or cable ranges that are connected and that have the following characteristics:

- Each network number or cable range within a domain is unique within that domain.
- Each domain is separated from another domain by a domain router.
- There is no physical or virtual connection between the two AppleTalk domains other than through a domain router.

## Understanding Domain Routers

The domain router uses split horizon across the entire domain, not just across an interface, which means that domain routers do not propagate routes learned from an interface in one domain back into that domain. Rather, domain routers propagate routes only to other domains.

## AppleTalk Interenterprise Routing Features

AppleTalk interenterprise routing provides the following features:

- Network remapping—Allows you to remap remote network numbers to resolve numbering conflicts with network numbers on the local network segment.
- Hop-count reduction—Allows the creation of larger internetworks. When you enable hop-count reduction, the hop count in a packet is set to 1 as it passes from one domain to another, therefore allowing you to circumvent the 15-hop limit imposed by DDP and RTMP when forwarding packets.
- Loop detection—Avoids having multiple routing table entries to the same remote network segment (domain). If the domain router detects a loop, it displays an error message on the domain router and shuts off domains. The presence of a loop implies that there is a connection between two separate domains that was not learned through any of the interfaces of the domain router.
- Fast switching—Has been implemented for networks that have been remapped or on which hop-count reduction has been configured.

## Redundant Paths Between Domains

Note that only one domain router can separate two domains. That is, you cannot have two or more domain routers to create redundant paths between domains. You can, however, establish redundant paths between domains by connecting them through more than one interface on the domain router that separates them. Figure 2 illustrates this configuration. In this figure, one domain router separates domains A and B. Two of the interfaces of the router are in Domain A (Ethernet interfaces 3 and 4), and three are in Domain B (Ethernet interfaces 0, 1, and 2), thus providing redundant connections between the domains. Figure 3 illustrates an improper configuration. This configuration will create adverse effects, because domains A and B are connected by two domain routers.

***Figure 2 Allowed Configuration of Domain Router Connecting Two Domains***

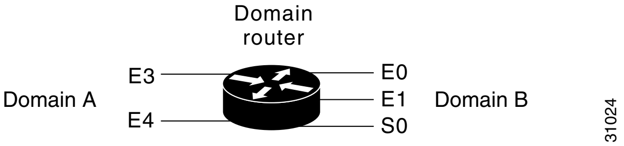
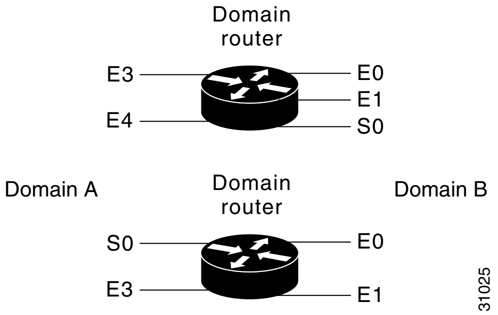


***Figure 3 Improper Configuration of Domain Routers Connecting Two Domains***



Currently, you can configure AppleTalk interenterprise routing only on routers that run RTMP.

## AppleTalk Interenterprise Routing Task List

To configure AppleTalk interenterprise routing, perform the tasks described in the following sections. At a minimum, you must enable AppleTalk interenterprise routing. The remaining tasks are optional.

- Enabling AppleTalk Interenterprise Routing (Required)
- Remapping Network Numbers (Optional)
- Controlling Hop Count (Optional)

After you assign AppleTalk interenterprise routing remapping, hop-count reduction, and loop-detection features to an AppleTalk domain, you can attribute those characteristics to a tunnel interface configured for AURP by assigning the AppleTalk domain group number to the AURP tunnel interface.

### Enabling AppleTalk Interenterprise Routing

To enable AppleTalk interenterprise routing, perform the following tasks:

- Enable AppleTalk interenterprise routing on the router. (Required)
- Enable AppleTalk interenterprise routing on an interface. (Required)

To enable AppleTalk interenterprise routing, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk domain domain-number name domain-name` | Creates a domain and assigns it a name and number. |

To enable AppleTalk interenterprise routing on an interface, use the following command in interface configuration mode:

| Command | Purpose |
|---|---|
| `Router(config-if)# appletalk domain-group domain-number` | Assigns a predefined domain number to an interface. |

For an example of how to configure AppleTalk interenterprise routing, see the "AppleTalk Interenterprise Routing Example" section at the end of this chapter.

### Remapping Network Numbers

When two AppleTalk networks are connected, a conflict can arise between network numbers or between cable ranges on one network and those on the other. You can avoid conflicts by remapping the network numbers or cable ranges of the remote network.

Each domain can have two mapping ranges to which to remap all incoming or outgoing network numbers or cable ranges.

To remap the network numbers or cable ranges on inbound packets, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk domain domain-number remap-range in cable-range` | Remaps packets inbound to the domain. |

To remap the network numbers or cable ranges on outbound packets, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk domain domain-number remap-range out cable-range` | Remaps packets outbound from the domain. |

### Controlling Hop Count

When you join AppleTalk network segments to create domains, the distance across the combined internetworks is likely to exceed 15 hops, which is the maximum number of hops supported by RTMP. You can extend the network topology by configuring the Cisco IOS software to reduce the hop-count value of packets that traverse it.

Reducing the hop-count value allows an AppleTalk router to control the hop-count field in DDP packets so as to ensure that the packet reaches its final AppleTalk destination. Hop-count reduction allows the router to bypass the limitation of 16 hops before aging out packets. This feature is supported only on access servers and routers configured for AppleTalk Enhanced IGRP.

To enable hop-count reduction, use the following command in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk domain domain-number hop-reduction` | Enables hop-count reduction. |

# Configuring AppleTalk over WANs

You can configure AppleTalk over dial-on-demand routing (DDR), Frame Relay, SMDS, and X.25 networks. For more information about dial-on-demand routing (DDR), refer to the *Cisco IOS Dial Technologies Configuration Guide.* For more information about Frame Relay, SMDS, and X.25, refer to the *Cisco IOS Wide-Area Networking Configuration Guide*.

## AppleTalk over DDR

To use AppleTalk over DDR, you must define AppleTalk static routes. You can configure the following two types of static routes:

- Static routes—These routes have absolute precedence (that is, they always override any dynamically learned routes).
- Floating static routes—These routes can be overridden by dynamically learned routes.

Be careful when assigning static routes. When links associated with these static routes are lost, traffic may stop being forwarded or traffic may be forwarded to a nonexistent destination, even though an alternative path might be available.

**Note** When you configure AppleTalk over DDR, the zone name assigned to the interface must be unique. It cannot be the same as a zone name assigned to a static route. If the zone names are not unique, the sequence of AppleTalk initialization and dialer operation will cause the DDR interface to go up and down.

### Configuring Static Routes

To add a static route for an extended or nonextended AppleTalk network, use one of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)# appletalk static cable-range cable-range to network.node zone zone-name` | Defines a static route on an extended AppleTalk network. |
| `Router(config)# appletalk static network network-number to network.node zone zone-name` | Defines a static route on a nonextended AppleTalk network. |

### Configuring Floating Static Routes

You can use a floating static route to create a path of last resort that is used only when no dynamic routing information is available. To avoid the possibility of a routing loop occurring, floating static routes by default are not redistributed into other dynamic protocols.

To add a floating static route for an extended or nonextended AppleTalk network, use one of the following commands in global configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk static cable-range`** *`cable-range`* **`to`** *`network`*`.`*`node`* **`floating zone`** *`zone-name`* | Defines a floating static route on an extended AppleTalk network. |
| `Router(config)#` **`appletalk static network`** *`network-number`* **`to`** *`network`*`.`*`node`* **`floating zone`** *`zone-name`* | Defines a floating static route on a nonextended AppleTalk network. |

For an example of how to configure AppleTalk over DDR, see the "AppleTalk over DDR Example" section at the end of this chapter.

## AppleTalk over X.25

For X.25, you can configure only a nonextended AppleTalk network. Logically, this network is the same as a LocalTalk network, because both are *always* nonextended networks. All AppleTalk nodes within an X.25 network must be configured with the same AppleTalk network number. Also, the network numbers and zone names on both sides of the serial link must be the same. When mapping the AppleTalk address to the X.121 address of the router with the **x25 map** command, include the keyword **broadcast** to simulate the AppleTalk broadcast capability. This keyword is necessary because X.25 does not support broadcasts, but AppleTalk does. The broadcast simulation operates as follows: If the broadcast flag is set, whenever a broadcast packet is sent, each X.121 address specified will receive it.

# Configuring AppleTalk Between LANs

For more information on Ethernet-emulated LANs and routing AppleTalk between them, refer to the "Configuring LAN Emulation" chapter of the *Cisco IOS Switching Services Configuration Guide*.

# Configuring AppleTalk Between VLANs

For more information on VLANs and routing AppleTalk between them over Inter-Switch Link (ISL) or IEEE 802.10, refer to the "Configuring Routing Between VLANs with ISL Encapsulation" and "Configuring Routing Between VLANs with IEEE 802.10 Encapsulation" chapters of the *Cisco IOS Switching Services Configuration Guide*.

# Monitoring and Maintaining the AppleTalk Network

The Cisco IOS software provides several commands that you can use to monitor and maintain an AppleTalk network. In addition, you can use network monitoring packages (such as *Inter•Poll* by Apple Computer) to verify that a router is configured and operating properly. Use the commands described in this section to monitor an AppleTalk network using both Cisco IOS software commands and network monitoring packages.

## Monitoring and Maintaining the AppleTalk Network Using Cisco IOS Software Commands

To monitor and maintain the AppleTalk network, use one or more of the following commands in EXEC configuration mode:

| Command | Purpose |
|---|---|
| `Router(config)#` **`appletalk pre-fdditalk`** | Enables recognition of pre-FDDITalk packets. |
| `Router>` **`clear appletalk arp`** `[`*`network.node`*`]` | Deletes entries from the AppleTalk ARP (AARP) table. |
| `Router>` **`clear appletalk neighbor`** `[`*`neighbor-address`*` \| `*`all`*`]` | Deletes entries from the neighbor table. |
| `Router>` **`clear appletalk route`** *`network`* | Deletes entries from the routing table. |
| `Router>` **`clear appletalk traffic`** | Resets AppleTalk traffic counters. |
| `Router>` **`ping appletalk`** *`network.node`* | Diagnoses basic AppleTalk network connectivity (user-level command). |
| `Router>` **`ping`** `[`**`appletalk`**`] [`*`network.node`*`]` | Diagnoses basic AppleTalk network connectivity (privileged command). |
| `Router#` **`show appletalk access-lists`** | Displays the AppleTalk access lists currently defined. |
| `Router#` **`show appletalk adjacent-routes`** | Displays the routes to networks that are directly connected or that are one hop away. |
| `Router#` **`show appletalk arp`** | Lists the entries in the AppleTalk ARP table. |
| `Router#` **`show appletalk aurp events`** | Displays pending events in the AppleTalk AURP update-events queue. |
| `Router#` **`show appletalk aurp topology`** | Displays entries in the AURP private path database. |
| `Router>` **`show appletalk cache`** | Displays the contents of the AppleTalk fast-switching cache. |
| `Router>` **`show appletalk domain`** `[`*`domain-number`*`]` | Displays domain-related information. |
| `Router>` **`show appletalk globals`** | Displays information about AppleTalk internetwork and other parameters of the router. |
| `Router#` **`show appletalk interface`** `[`**`brief`**`] [`*`type number`*`]` | Displays AppleTalk-related interface settings. |
| `Router>` **`show appletalk macip-clients`** | Displays the status of all known MacIP clients. |
| `Router>` **`show appletalk macip-servers`** | Displays the status of MacIP servers of a device. |
| `Router>` **`show appletalk macip-traffic`** | Displays statistics about MacIP traffic. |
| `Router#` **`show appletalk name-cache`** | Displays a list of NBP services offered by nearby routers and by other devices that support NBP. |
| `Router>` **`show appletalk nbp`** | Displays the contents of the NBP name registration table. |

| Command | Purpose |
|---|---|
| `Router>` **`show appletalk neighbors`** `[`*`neighbor-address`*`]` | Displays information about the AppleTalk routers directly connected to any network to which the router is directly connected. |
| `Router>` **`show appletalk remap`** `[`**`domain`** *`domain-number`* `[{`**`in`** `\|` **`out`**`} [{`**`to`** `\|` **`from`**`}` *`domain-network`*`]]]` | Displays domain remapping information. |
| `Router>` **`show appletalk route`** `[`*`network`* `\|` *`type number`*`]` | Displays the contents of the AppleTalk routing table. |
| `Router#` **`show appletalk sockets`** `[`*`socket-number`*`]` | Displays the process-level operations in all sockets in an interface. |
| `Router>` **`show appletalk static`** | Displays the defined static routes. |
| `Router>` **`show appletalk traffic`** | Displays the statistics about AppleTalk protocol traffic, including MacIP traffic. |
| `Router>` **`show appletalk zone`** `[`*`zone-name`*`]` | Displays the contents of the zone information table. |
| `Router#` **`test appletalk`** | Enters test mode to test NBP protocols. |

## Monitoring the AppleTalk Network Using Network Monitoring Packages

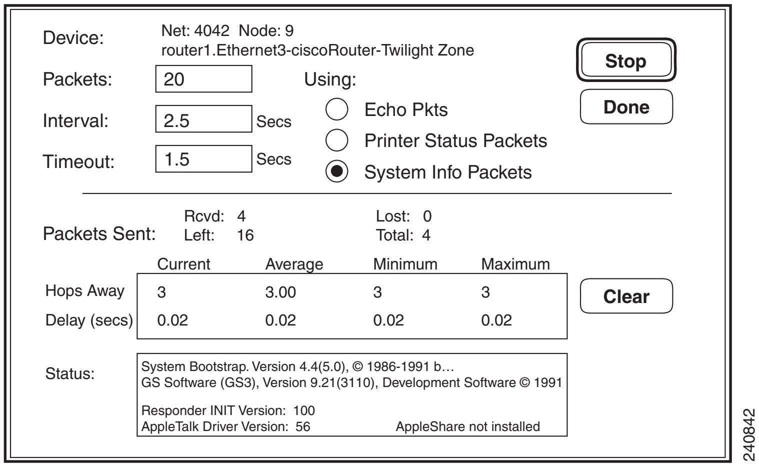
The Cisco IOS software supports network monitoring packages (such as *Inter•Poll* by Apple Computer*)*, which are tools that use the AppleTalk responder and listener for verifying the configuration and operation of a router. The software answers AppleTalk *responder* request packets. These request packets are received by the *listener*, which is installed on the AppleTalk interface name registration socket. The responder request packets include the bootstrap firmware version string, followed by the operating software version string. These strings are displayed in the Macintosh system version and the Macintosh printer driver version fields, respectively, and in applications such as *Inter•Poll* by Apple Computer. The response packet contains strings similar to those displayed by the **show version** EXEC command.

The Cisco IOS software returns the following information in response to responder request packets:

- System bootstrap version (ROM version)
- Software version
- AppleTalk version (always version 56, which is the first Apple Macintosh version that contained AppleTalk Phase 2 support)
- AppleTalk responder version (always version 100, which indicates support of Version 1.0 responder packets)
- AppleShare status (reported as “not installed”)

Figure 4 illustrates a typical output display for *Inter•Poll* that lists this information.

*Figure 4 Inter•Poll Output*



Device: Net: 4042 Node: 9
router1.Ethernet3-ciscoRouter-Twilight Zone

Packets: 20

Interval: 2.5 Secs

Timeout: 1.5 Secs

Using:
○ Echo Pkts
○ Printer Status Packets
● System Info Packets

Packets Sent: Rcvd: 4 Left: 16 Lost: 0 Total: 4

| | Current | Average | Minimum | Maximum |
|---|---|---|---|---|
| Hops Away | 3 | 3.00 | 3 | 3 |
| Delay (secs) | 0.02 | 0.02 | 0.02 | 0.02 |

Status:
System Bootstrap. Version 4.4(5.0), © 1986-1991 b…
GS Software (GS3), Version 9.21(3110), Development Software © 1991

Responder INIT Version: 100
AppleTalk Driver Version: 56 AppleShare not installed

# AppleTalk Configuration Examples

To help you configure AppleTalk routing, use the configuration examples in the following sections:

- Extended AppleTalk Network Example
- Nonextended AppleTalk Network Example
- Nonextended Network in Discovery Mode Example
- Transition Mode Example
- AppleTalk Access List Examples
- Hiding and Sharing Resources with Access List Examples
- GZL and ZIP Reply Filter Examples
- AppleTalk Interenterprise Routing over AURP Example
- SNMP Example
- MacIP Examples
- IPTalk Example
- AppleTalk Control Protocol Example
- Proxy Network Number Example
- AppleTalk Interenterprise Routing Example
- AppleTalk over DDR Example
- AppleTalk Control Protocol for PPP Example

## Extended AppleTalk Network Example

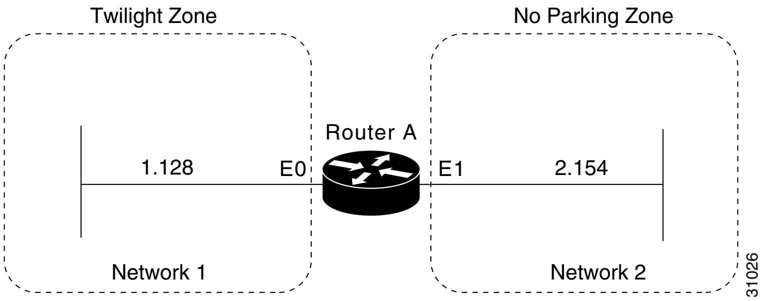
The following example shows how to configure an extended AppleTalk network. It defines the zones Accounting and Personnel. The cable range of 1 allows compatibility with nonextended AppleTalk networks.

```
appletalk routing
interface ethernet 0
 appletalk cable-range 69-69 69.128
 appletalk zone Accounting
 appletalk zone Personnel
```

# Nonextended AppleTalk Network Example

The following example shows how to configure a nonextended AppleTalk network that allows routing between two Ethernet networks. Ethernet interface 0 is connected to Network 1 at node 128, and Ethernet interface 1 is connected to Network 2 at node 154. Network 1 is in the Twilight zone, and Network 2 is in the No Parking zone. See Figure 5.

*Figure 5 Nonextended AppleTalk Routing Between Two Ethernet Networks*

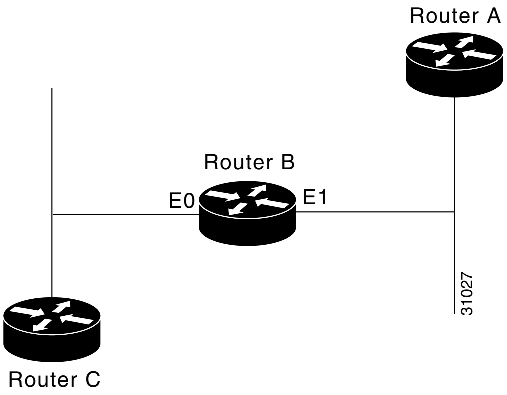


```
appletalk routing
!
interface ethernet 0
appletalk address 1.128
appletalk zone Twilight
!
interface ethernet 1
 appletalk address 2.154
 appletalk zone No Parking
```

# Nonextended Network in Discovery Mode Example

The following example shows how to configure a nonextended network in discovery mode. There are seed routers on both networks to provide the zone and network number information to the interfaces when they start. Router A supplies configuration information for Ethernet interface 1, and Router C supplies configuration information for Ethernet interface 0. See Figure 6.

*Figure 6 Routing in Discovery Mode*



The following example shows how to configure this nonextended network in discovery mode:

```
appletalk routing
!
interface ethernet 0
 appletalk address 0.0
!
interface ethernet 1
 appletalk address 0.0
```

# AppleTalk Access List Examples

Our implementation of AppleTalk provides several methods using access lists to control access to AppleTalk networks. The following sections show these methods and different approaches in applying access lists.

## Defining an Access List to Filter Data Packets Example

The following commands create access list 601:

```
! Permit packets to be routed from network 55.
access-list 601 permit network 55

! Permit packets to be routed from network 500.
access-list 601 permit network 500

! Permit packets to be routed from networks 900 through 950.
access-list 601 permit cable-range 900-950

! Do not permit packets to be routed from networks 970 through 990.
access-list 601 deny includes 970-990

! Do not permit packets to be routed from networks 991 through 995.
access-list 601 permit within 991-995

! Deny routing to any network and cable range not specifically enumerated.
access-list 601 deny other-access
```

The following example shows how to use access list 601 to filter data packets by applying an interface (for example, Ethernet interface 0):

```
appletalk routing
interface ethernet 0
 appletalk cable-range 50-50
 appletalk zone No Parking
 appletalk access-group 601 out
```

The following examples show how Ethernet interface 0 would handle outgoing data packets:

- Packets sourced from cable range 50–50 are permitted.
- Packets sourced from any network in the cable range 972–980 are denied because they explicitly match the **access-list deny includes 970-990** command.

## Defining an Access List to Filter Incoming Routing Table Updates Example

The following example shows how to create access list 602. This example shows how packets are processed by access lists; you cannot create such a redundant access list.

```
access-list 602 permit network 55
access-list 602 permit cable 55-55
access-list 602 permit includes 55-55
access-list 602 permit within 55-55
```

The following example shows how to use this access list to filter routing table updates received on Ethernet interface 0:

```
appletalk routing
interface ethernet 0
 appletalk cable-range 55-55
 appletalk zone No Parking
 appletalk distribute-list 602 in
```

The following tables show the process for accepting or rejecting routing update information. If the outcome of a test is *true*, the condition passes the access list specification and the **distribute-list** command specification is then applied.

Routing updates that contain network 55 would be processed as follows:

| Access List Command | Outcome of Test |
|---|---|
| access-list 602 permit network 55 | True |
| access-list 602 permit cable range 55–55 | False |
| access-list 602 permit includes 55–55 | True |
| access-list 602 permit within 55–55 | True |

Routing updates that contain cable range 55–55 would be processed as follows:

| Access List Command | Outcome of Test |
|---|---|
| access-list 602 permit network 55 | False |
| access-list 602 permit cable range 55–55 | True |
| access-list 602 permit includes 55–55 | True |
| access-list 602 permit within 55–55 | True |

Routing updates that contain cable range 55–56 would be processed as follows:

| Access List Command | Outcome of Test |
|---|---|
| access-list 602 permit network 55 | False |
| access-list 602 permit cable-range 55–55 | False |
| access-list 602 permit includes 55–55 | True |
| access-list 602 permit within 55–55 | False |

## Comparison of Alternative Segmentation Solutions

With the flexibility allowed by our access list implementation, determining the optimal method to segment an AppleTalk environment using access control lists can be unclear. The following scenario and configuration examples illustrate two solutions to a particular problem, and point out the inherent advantages of using AppleTalk-style access lists.

Consider a situation in which a company wants to permit customers to have direct access to several corporate file servers. Access is to be permitted to all devices in the zones named MIS and Corporate, but access is restricted to the Engineering zone because the file servers in these zones contain sensitive information. The solution is to create the appropriate access lists to enforce these access policies.

The AppleTalk internetwork of the company consists of the following networks and zones:

| Zone | Network Number or Cable Range |
|---|---|
| Engineering | 69–69<br>3<br>4160–4160<br>15 |
| MIS | 666–777 |
| Corporate | 70–70<br>55<br>51004<br>4262–4262 |
| World | 88–88<br>9<br>9000–9999 (multiple networks exist in this range) |

The router named Gatekeeper is placed between the World zone and the various company-specific zones. An arbitrary number of routers can be on either side of Gatekeeper. An Ethernet backbone exists on each side of Gatekeeper, connecting these other routers to Gatekeeper. On the router Gatekeeper, Ethernet interface 0 connects to the World backbone and Ethernet interface 1 connects to the Corporate backbone.

For the purposes of this configuration, assume Gatekeeper is the only router that needs any access list configuration. There are two solutions, depending on the level of security desired.

The following example shows a minimal configuration, in which the Engineering zone is secured, but all other zones are publicly accessible:

```
appletalk routing
access-list 603 deny zone Engineering
access-list 603 permit additional-zones
```

```
access-list 603 permit other-access

interface ethernet 0
appletalk network 3
 appletalk distribute-list 603 out
 appletalk access-group 603
```

The following example shows a more comprehensive configuration, in which the Corporate and MIS zones are public and all other zones are secured:

```
appletalk routing
access-list 603 permit zone Corporate
access-list 603 permit zone MIS
access-list 603 deny additional-zones
access-list 603 permit other-access

interface ethernet 0
appletalk network 3
 appletalk distribute-list 603 out
 appletalk access 603
```

Both configurations satisfy the basic goal of isolating the Engineering servers, but the second example will continue to be secure when more zones are added.

## Defining an Access List to Filter NBP Packets Example

The following example shows how to add entries to access list number 607 to allow forwarding of NBP packets from specific sources and deny forwarding of NBP packets from all other sources. The first command adds an entry that allows NBP packets from all printers of type LaserWriter. The second command adds an entry that allows NBP packets from all AppleTalk file servers of type AFPServer. The third command adds an entry that allows NBP packets from all applications called HotShotPaint. For example, an application might have a **zone** name of Accounting and an application might have a **zone** name of Engineering, both having the object name of HotShotPaint. NBP packets forwarded from both applications will be allowed.

The final **access-list other-nbps** command denies forwarding of NBP packets from all other sources.

```
access-list 607 permit nbp 1 type LaserWriter
access-list 607 permit nbp 2 type AFPServer
access-list 607 permit nbp 3 object HotShotPaint
access-list 607 deny other-nbps
```

The following example shows how to use this access list to filter inbound NBP packets on Ethernet interface 0:

```
appletalk routing
interface ethernet 0
 appletalk cable-range 55-55
 appletalk zone No Parking
 appletalk access-group 607 in
```

The following example shows how to add entries to access list number 608 to deny forwarding of NBP packets from two specific servers whose fully qualified NBP names are specified. It permits forwarding of NBP packets from all other sources.

```
access-list 608 deny nbp 1 object ServerA
access-list 608 deny nbp 1 type AFPServer
access-list 608 deny nbp 1 zone Bld3
access-list 608 deny nbp 2 object ServerB
access-list 608 deny nbp 2 type AFPServer
access-list 608 deny nbp 2 zone Bld3
```

```
access-list 608 permit other-nbps
access-list 608 permit other-access
```

The following example shows how to use this access list to filter NBP packets on Ethernet interface 0:

```
appletalk routing
interface ethernet 0
 appletalk cable-range 55-55
 appletalk zone No Parking
 appletalk access-group 608 in
```

**Note** Prior to Cisco IOS Release 11.2 F, all NBP access lists were applied to inbound interfaces by default. When Cisco IOS Release 11.2 F or later software is used, the default interface direction for all access lists, including NBP access lists, is outbound. In order to retain the inbound direction of access lists created with previous Cisco IOS software releases, you must specify an inbound interface for all NBP access lists using the **appletalk access-group** command.

The following example shows how to create an access list that denies forwarding of the following:

- All NBP Lookup Reply packets
- NBP packets from the server named Bob’s Server
- Packets from all AppleTalk file servers of type AFPServer
- All NBP Lookup Reply packets that contain the specified named entities belonging to the zone twilight:

```
access-list 600 deny nbp 1 LkReply
access-list 600 deny nbp 1 object Bob’s Server
access-list 600 deny nbp 1 type AFPServer
access-list 600 deny nbp 1 zone twilight
access-list 600 permit other-nbps
```

There may be a case where a fully qualified filter for Bob’s Server:AFPServer@twilight will not work for an NBP Lookup Reply in response to a Lookup generated by the Chooser application. This case would occur because the Lookup Request is sent as =:AFPServer@twilight, and the Lookup Reply from Bob’s Server comes back as Bob’s Server:AFPServer@*.

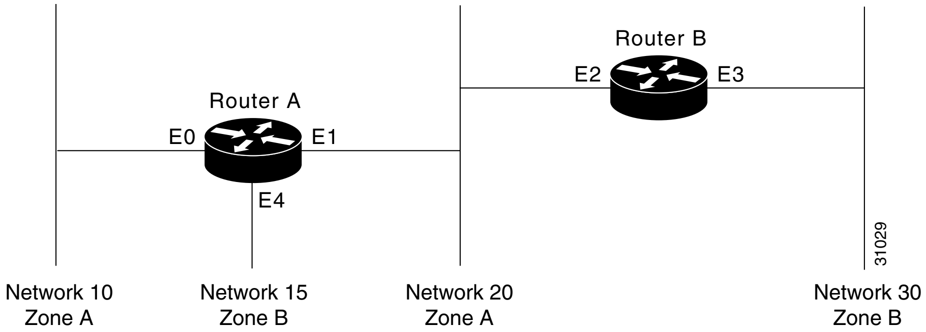
The following example shows how to create an access list to filter a Lookup Reply generated by Bob’s Server to a request by the Chooser application:

```
access-list 609 deny nbp 1 LkReply
access-list 609 deny nbp 1 object Bob’s Server
access-list 609 deny nbp 1 type AFPServer
access-list 609 permit other-nbps
access-list 609 permit other-access
```

## Configuring Partial Zone Advertisement Example

Figure 7 illustrates a configuration in which you might want to allow partial advertisement of a particular zone.

*Figure 7 Example Topology of Partially Obscured Zone*



Assume that Router B includes a router-update filter (applied with the **appletalk distribute-list** interface configuration command) on the Ethernet interface 3 that does not accept routing table updates from network 10, nor does it send routing table updates to that network.

```
access-list 612 deny network 10
access-list 612 permit other-access
interface ethernet 3
 appletalk distribute-list 612 out
 appletalk distribute-list 612 in
```

For Network 30, normal (default) behavior would be for Network 10 and Network 20 to be eliminated from any routing updates sent, although Network 15 would be included in routing updates (same zone as Network 30). Using the **appletalk permit-partial-zones** global configuration command has the following effects:

- If the **appletalk permit-partial-zones** command is enabled, the routing updates exclude Network 10, but *include* Network 15 and Network 20.
- If the **no appletalk permit-partial-zones** command is enabled, the routing updates exclude both Network 10 and Network 20, but still include Network 15. This configuration is generally considered the preferred behavior and is the default.

Table 4 summarizes the associations between the networks shown in Figure 7. Table 5 details the effects of enabling and disabling partial-zone advertisement with the **appletalk permit-partial-zones** global configuration command.

*Table 4 Zone and Interface Associations for Partial Zone Advertisement Example*

| Network | Network 10 | Network 15 | Network 20 | Network 30 |
|---|---|---|---|---|
| Zone | A | B | A | B |
| Interfaces | Ethernet 0 | Ethernet 4 | Ethernet 1<br>Ethernet 2 | Ethernet 3 |

*Table 5 Partial Zone Advertisement Control on Network 30*

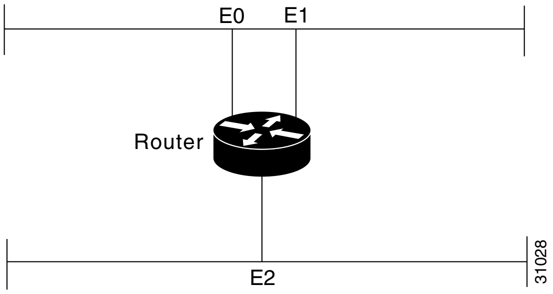
| Command Condition | Network 10 | Network 15 | Network 20 | Network 30 |
|---|---|---|---|---|
| Enabled | Not advertised on Network 30 | Advertised on Network 30 | Advertised on Network 30 | — |
| Disabled | Not advertised on Network 30 | Advertised on Network 30 | Not advertised on Network 30 | — |

# Transition Mode Example

When in transition mode, the Cisco IOS software can route packets between extended and nonextended AppleTalk networks that exist on the same cable.

To configure transition mode, you must have two ports connected to the same physical cable. One port is configured as a nonextended AppleTalk network, and the other is configured as an extended AppleTalk network. Both ports must have unique network numbers, because they are two separate networks. Figure 8 shows an example of the topology of this configuration.

*Figure 8 Transition Mode Topology and Configuration*



The following example shows how to configure the network shown in Figure 8. Note that networks 2-2 and 4-4 must have a cable range of 1 and a single zone in their zone lists. These parameters are required to maintain compatibility with the nonextended network, network 3.

```
! This is an extended network.
interface ethernet 0
 appletalk cable-range 2-2
 appletalk zone No Parking
!
! This is a nonextended network.
interface ethernet 1
 appletalk address 3.128
 appletalk zone Twilight
!
! This is an extended network.
interface ethernet 2
 appletalk cable-range 4-4
 appletalk zone Do Not Enter
```

# Hiding and Sharing Resources with Access List Examples

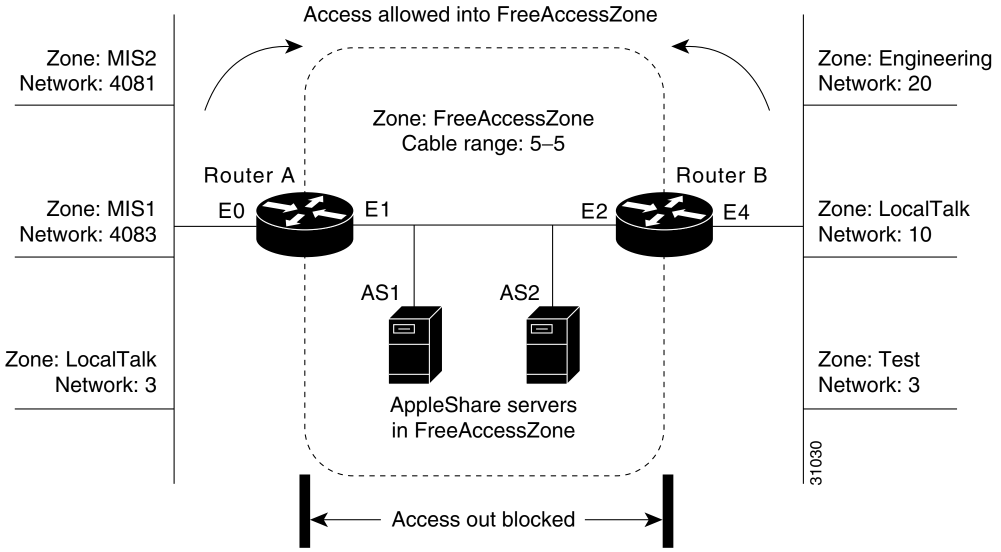
The following examples show how to use AppleTalk access lists to manage access to certain resources.

## Establishing a Free-Trade Zone Example

The goal of the configuration shown in Figure 9 is to allow all users on all the networks connected to Router A and Router B to be able to access the AppleShare servers AS1 and AS2 in the zone FreeAccessZone. A second requirement is to block cross access through this zone. In other words, users in the zones MIS1, MIS2, and LocalTalk (which are connected to Ethernet interface 0 on Router A) are not allowed access to any of the resources on networks connected to Ethernet interface 4 on Router B.

Similarly, users in the zones Engineering, Test, and LocalTalk (which are connected to Ethernet interface 4 on Router B, interface E4) are not allowed access to any of the resources on networks connected to Ethernet interface 0 on Router A.

***Figure 9 Controlling Access to a Common AppleTalk Network***



**Note** Although there are networks that share the same number on interfaces E0 and E4 and there are zones that have the same name, none have the same network number and zone specification (except FreeAccessZone). The two routers do *not* broadcast information about these networks through FreeAccessZone. The routers only broadcast the cable range 5–5. As configured, FreeAccessZone sees only itself. However, because no other limitations have been placed on advertisements, the FreeAccessZone range of 5–5 propagates out to the networks attached to E0 (Router A) and E4 (Router B); thus, resources in FreeAccessZone are made accessible to users on all those networks.

The following examples configure Router A and Router B for access control illustrated in Figure 9. You must configure only Ethernet interface 1 on Router A and Ethernet interface 2 on Router B to provide the desired access.

### Configuration for Router A

```
appletalk routing
!
interface ethernet 1
 appletalk cable-range 5-5
 appletalk zone FreeAccessZone
 appletalk free-trade-zone
```

### Configuration for Router B

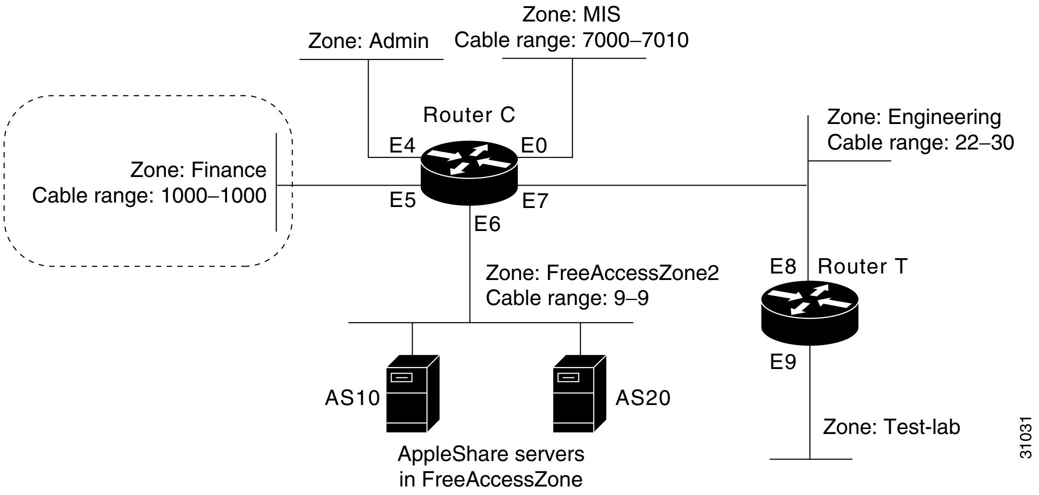
```
appletalk routing
!
interface ethernet 2
 appletalk cable-range 5-5
 appletalk zone FreeAccessZone
 appletalk free-trade-zone
```

When configuring both routers, you need not define any access lists to prevent users on networks connected to Router A from accessing resources on networks connected to Router B, and vice versa. The **appletalk free-trade-zone** interface configuration command implements the necessary restrictions.

## Restricting Resource Availability Example

In the preceding example, shared-resource access was granted to all users in the various AppleTalk zones connected to the two routers. At the same time, access between resources on either side of the common zone was completely denied. There might be instances where a greater degree of control is required—possibly where resources in some zones are to be allowed access to resources in certain other zones, but are denied access to other specific zones. Figure 10 illustrates such a situation.

***Figure 10 Controlling Resource Access Among Multiple AppleTalk Zones***



The following are the objectives of the configuration in Figure 10:

- Users in zones Engineering (E7) and MIS (E0) are to be allowed free access to each other.
- All users in all zones are to be allowed access to FreeAccessZone2 (E6).
- No users in any zone, with the exception of users in Finance, are to be allowed access to resources in Finance.

The following example shows how to meet these specifications:

```
access-list 609 permit cable 9-9
access-list 609 deny other-access
!
access-list 610 permit zone Finance
access-list 610 permit zone FreeAccessZone2
access-list 610 deny additional-zones
!
access-list 611 deny cable-range 1000-1000
access-list 611 deny cable-range 9-9
access-list 611 permit cable-range 7000-7010
access-list 611 permit cable-range 22-30
```

The effects of these access lists are as follows:

- Access list 609 is intended to be used to allow access to resources on FreeAccessZone2.
- Access list 610 is intended to be used to control access in and out of the zone Finance.

- Access list 611 is intended to be used to accommodate the requirement to allow users in zones Engineering and MIS to mutually access network resources.

### Configuration for Ethernet Interface 0

Ethernet interface 0 is associated with the MIS zone. The following example shows how to configure this interface:

```
interface ethernet 0
 appletalk cable-range 7000-7010
 appletalk zone MIS
 appletalk distribute-list 611 out
 appletalk distribute-list 611 in
```

Specifying access list 611 results in the following filtering:

- Advertisements of Finance are blocked.
- Advertisements between Engineering and MIS are allowed.

### Configuration for Ethernet Interface 5

Ethernet interface 5 is associated with the Finance zone. The following example shows how to configure this interface:

```
interface ethernet 5
 appletalk cable-range 1000-1000
 appletalk zone Finance
 appletalk distribute-list 610 out
 appletalk access-group 610
```

The effects of these access lists are as follows:

- With the **appletalk distribute-list out** interface configuration command, Finance is limited to accessing Finance and FreeAccessZone2 only.
- The **appletalk access-group** interface configuration command filters packet traffic. Thus, it blocks access to any devices in Finance from outside of this zone.

### Configuration for Ethernet Interface 6

Ethernet interface 6 is associated with the FreeAccessZone2 zone. The following example shows how to configure this interface:

```
interface ethernet 6
 appletalk cable 9-9
 appletalk zone FreeAccessZone2
 appletalk distribute-list 609 out
 appletalk distribute-list 609 in
```

### Configuration for Ethernet Interface 7

Ethernet interface 7 is associated with the Engineering zone. The configuration for this interface mirrors that for Ethernet interface 0, because the users in both the MIS and Engineering zones must have access to resources from each other. The following example shows how to configure Ethernet interface 7:

```
interface ethernet 7
 appletalk cable-range 22-30
 appletalk zone Engineering
 appletalk distribute-list 611 out
 appletalk distribute-list 611 in
```

### Implicit Configuration of the Admin and Test-Lab Zones

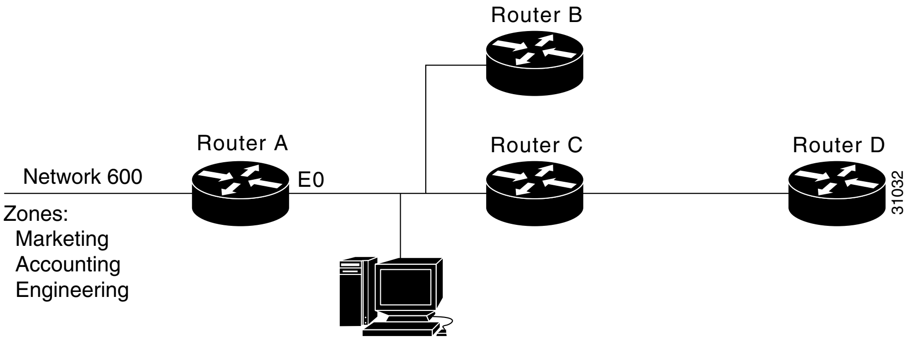
Omitted from the configuration example in Figure 10 are any specific configuration commands pertaining to the zones Test-Lab (Ethernet interface 9 on Router T) and Admin (Ethernet interface 4 on Router C). No configuration is done for these zones because there are no requirements relating to them listed in the original objectives. The following access control is implicitly handled with the assignment of the stated access lists:

- Users in the Admin zone can see the Finance zone, but cannot see resources in that zone. However, as for all zones, resources in FreeAccessZone2 are available, but none of the users in any of the other zones can access resources in Admin.
- In the absence of the assignment of access lists on Router T, users in Test-Lab can access the resources in the FreeAccessZone2 and Engineering zones. With the exception of Engineering, no other zones can access resources in Test-Lab.

## GZL and ZIP Reply Filter Examples

The following examples show how to configure GZL and ZIP reply filters and the differences between these two types of filters. Both examples use the configuration shown in Figure 11.

***Figure 11 Sample Topology for GZL and ZIP Reply Filters***



Both GZL and ZIP reply filters control the zones that can be seen on a network segment. GZL filters control which zones can be seen by Macintoshes on local network segments. These filters have no effect on adjacent routers. In order for GZL filters to work properly, all routers on the local segment must be configured with the same access list.

ZIP reply filters control which zones can be seen by adjacent routers and by all routers downstream from adjacent routers. You can use these filters to hide zones from all Macintoshes on all networks on adjacent routers and from all their downstream routers.

Using the configuration shown in Figure 11, you would use a GZL filter to prevent the Macintosh on the Ethernet 0 network segment from viewing the zones Engineering and Accounting on network 600. These zones would not be visible via the Macintosh Chooser. The following example shows how to configure Router A:

```
access-list 600 deny zone Engineering
access-list 600 deny zone Accounting
access-list 600 permit additional-zones
access-list 600 permit other-access
!
interface ethernet 0
 appletalk getzonelist-filter 600
```

Again using the configuration shown in Figure 11, you would use a ZIP reply filter to hide the Engineering and Accounting zones from Router B and Router C. This filter would also hide the zones from Router D, which is downstream from Router C. The effect of this filter is that when these routers request the names of zones on network 600, the zones names Engineering and Accounting will not be returned.

```
access-list 600 deny zone Engineering
access-list 600 deny zone Accounting
access-list 600 permit additional-zones
access-list 600 permit other-access
!
interface ethernet 0
 appletalk zip-reply-filter 600
```

## AppleTalk Interenterprise Routing over AURP Example

After you configure an AppleTalk domain for AppleTalk interenterprise features, you can apply the features to a tunnel interface configured for AURP by assigning the domain number to the interface.

The following example shows how to define tunnel interface 0 and configure it for AURP. Then, it shows how to apply the features configured for domain 1 to tunnel interface 1 by assigning the AppleTalk domain group 1 to the tunnel interface.

```
appletalk domain 1 name France
appletalk domain 1 remap-range in 10000-19999
appletalk domain 1 remap-range out 200-299
!
interface Tunnel 0
 tunnel source ethernet 0
 tunnel destination 172.19.1.17
 tunnel mode aurp
 appletalk protocol aurp
 appletalk domain-group 1
```

## SNMP Example

The following example shows how to activate SNMP and AppleTalk:

```
! Disable SNMP on the router.
no snmp-server
!
! Enable AppleTalk routing and event logging on the router.
appletalk routing
appletalk event-logging
!
! Configure IP and AppleTalk on Ethernet interface 0.
interface Ethernet 0
ip address 131.108.29.291 255.255.255.0
 appletalk cable-range 29-29 29.180
 appletalk zone MarketingA1
!
! Enable SNMP on the router.
snmp-server community MarketingA2 RW
snmp-server trap-authentication
snmp server host 131.108.2.160 MarketingA2
```

## MacIP Examples

The following example shows how to configure MacIP support for dynamically addressed MacIP clients with dynamically allocated IP addresses in the range 172.18.0.2 to 172.18.0.10:

```
! Specify server address and zone.
appletalk macip server 172.18.0.1 zone Marketing
!
! Specify dynamically addressed clients.
appletalk macip dynamic 172.18.0.2 172.18.0.10 zone Marketing
!
! Assign the address and subnet mask for Ethernet interface 0.
interface ethernet 0
ip address 172.18.0.2 255.255.255.0
!
! Enable AppleTalk routing.
appletalk routing
!
interface ethernet 0
 appletalk cable range 69-69 69.128
 appletalk zone Marketing
```

The following example shows how to configure MacIP support for MacIP clients with statically allocated IP addresses:

```
! Specify the server address and zone.
appletalk macip server 172.18.0.1 zone Marketing
!
! Specify statically addressed clients.
appletalk macip static 172.18.0.11 172.18.0.20 zone Marketing
appletalk macip static 172.18.0.31 zone Marketing
appletalk macip static 172.18.0.41 zone Marketing
appletalk macip static 172.18.0.49 zone Marketing
!
! Assign the address and subnet mask for Ethernet interface 0.
interface ethernet 0
ip address 172.18.0.1 255.255.255.0
!
! Enable AppleTalk routing.
appletalk routing
!
interface ethernet 0
 appletalk cable range 69-69 69.128
 appletalk zone Marketing
```

## IPTalk Example

This section describes how to set up UNIX-based systems and our Cisco IOS software to use CAP IPTalk and other IPTalk implementations.

The following procedure outlines the basic steps for setting up our software and UNIX hosts for operation using IPTalk implementations.

**Note** This procedure does not provide full instructions about how to install CAP on the UNIX system. However, it does address the requirements for setting up the configuration file of the UNIX system that defines addresses and other network information. Generally, this is the only file that relies on the address and configuration information of the router. Refer to your UNIX system and CAP software manuals for information about building the CAP software and setting up the UNIX startup scripts.

**Step 1** Enable AppleTalk routing on all the routers that will use IPTalk and any routers between these routers.

**Step 2** Enable IP routing on the interfaces that will communicate with the UNIX system. (Refer to the *Cisco IOS IP and IP Routing Configuration Guide* for more information about configuring IP.) These interfaces must be on *the same subnet* as the UNIX system. Also, ensure that IP is enabled on the UNIX system.

**Step 3** Allocate an AppleTalk network number for IPTalk. You need a separate AppleTalk network number for each IP subnet that is to run IPTalk.

You can have a number of UNIX machines on the same subnet. They all use the same AppleTalk network number for IPTalk. However, they must have their own individual node identifiers.

It is possible for the same router to have IPTalk enabled on several interfaces. Each interface must have a different AppleTalk network number allocated to IPTalk, because each interface will be using a different IP subnet.

**Step 4** Determine the CAP format of the AppleTalk network number. The CAP software is based on an older AppleTalk convention that expresses AppleTalk network numbers as two octets (decimal numbers from 0 to 255) separated by a dot. The current AppleTalk convention uses decimal numbers from 1 to 65,279. Use the following formula to convert between the two:

CAP format: *x.y*
Apple format: *d*

To convert from AppleTalk to CAP:
$x = d/256$ (/ represents truncating integer division)
$y = d\%256$ (% represents the remainder of the division)

To convert from CAP to AppleTalk:
$d = x * 256 + y$

*Example:*
AppleTalk format: 14087
CAP format: 55.7

**Step 5** Choose a zone name for IPTalk. No special constraints are placed on zone name choices. You can use the same zone name for several networks, and you can combine IPTalk and normal AppleTalk networks in the same zone.

**Step 6** Decide which UDP ports to use for IPTalk. The default is to use ports beginning with 768. Thus, RTMP uses port 769, NBP port 770, and so on. These are the original AppleTalk ports, and their numbers are hardcoded into older versions of CAP. The only problem with using them is that they are not officially assigned by the Internet's NIC, which has assigned a set of UDP ports beginning with 200. Thus, other applications could use them, possibly causing conflicts—although this is unlikely. With CAP releases 5.0 and later, you can configure CAP to use the officially allocated ports. If you do so, RTMP will use port 201, NBP port 202, and so on. Whichever ports you use, you must configure both CAP and the router to use the same ones.

The following example shows how to enable IPTalk on each interface of the router as required:

```
appletalk routing
!
interface ethernet 0
 ip address 172.16.7.22 255.255.255.0
 appletalk cable 1792-1792 1792.22
 appletalk zone MIS-Development
interface Tunnel0
tunnel source Ethernet0
tunnel mode iptalk
appletalk iptalk 14087 MIS-UNIX
```

In this example, AppleTalk routing is enabled on the interface in the following two ways:

- Via EtherTalk phase 2, using the cable range 1792–1792 and the zone MIS-Development
- Via IPTalk, using the network number 14087 and the zone MIS-UNIX

**Note** The IPTalk node identifier is chosen automatically, based on the IP address. It is normally the host number portion of the IP address. For example, with an IP address of 172.16.7.22 and a subnet mask of 255.255.255.0, the host number is 22. Thus, the IPTalk node identifier would be 22. If the IP host number is larger than 255, the low-order 8 bits are used, although fewer than 8 bits may be available, depending on the IP subnet mask. If the mask leaves fewer bits, the node number will be quietly truncated. Be sure to use a node address that is compatible with the subnet mask. In any event, you may experience problems when using IPTalk with host numbers larger than 255.

If you choose to use the official UDP ports (those beginning with 200), include the following global configuration command in your configuration:

```
appletalk iptalk-baseport 200
```

**Step 7** Configure each UNIX host with a network number, zone name, and router.

The following example shows the contents of the /etc/atalk.local file from a UNIX system with the IP address 172.19.7.26 and a network mask of 255.255.255.0:

```
# IPTalk on net 172.19.7.0:
# mynet mynode myzone
55.7  26      MIS-UNIX
# bridgenet bridgenode bridgeIP
55.7  22      172.19.7.22
```

The first noncommented line defines the address of the UNIX system, and the second noncommented line defines the address of the router. In both cases, the first number is 55.7, which is the AppleTalk network number (in CAP format) for use by IPTalk. The second number is the AppleTalk node identifier, which must be the same as the IP host number. The last number on the first line is the zone name, and on the second line it is the IP address of the router.

Note the following about the entries in the /etc/atalk.local file:

- The AppleTalk network number in the first column in both lines must agree with the AppleTalk network number used in the **appletalk iptalk** command. However, in the /etc/atalk.local file, the number must be in the CAP format, while in the configuration command, it must be in the Apple format.
- The host number in the second column in both lines must agree with the IP host number of the corresponding system. That is, on the first line it must be the IP host number of the UNIX machine, and on the second line it must be the IP host number for the router.
- The zone name in the third column on the first line must agree with the zone name used in the **appletalk iptalk** command.
- The IP address in the third column of the second line must be the IP address of the router.

**Step 8** Ensure that your CAP software is using the same UDP port numbers as the router. Currently, the CAP default is the same as the router default, which is port numbers beginning with 768. If you want to use this default, you need not take any further action. However, if you want to use the official UDP port numbers (port numbers beginning with 200), ensure that you have included the following command in your configuration:

```
appletalk iptalk-baseport 200
```

**Step 9** On the UNIX system, add the following lines to the /etc/services file:

```
at-rtmp      201/udp
at-nbp       202/udp
at-3         203/udp
at-echo      204/udp
at-5         205/udp
at-zis       206/udp
at-7         207/udp
at-8         208/udp
```

If you are using Network Information Services (NIS), previously known as the Yellow Pages, remember to do a *make* in /var/yp after changing /etc/services. If you are using the default ports (those starting with 768), you need not modify /etc/services.

# AppleTalk Control Protocol Example

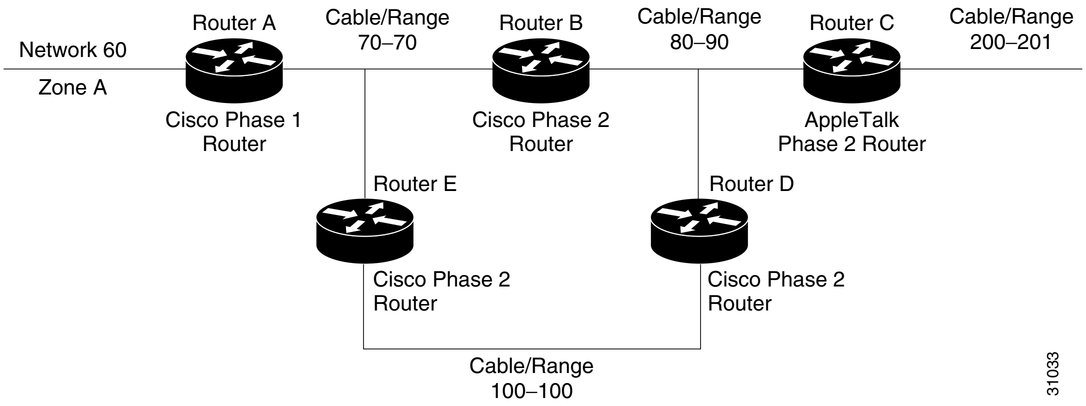
The following example shows how to set up a router to accept AppleTalk client requests on asynchronous interface 1 and create virtual network number 3 and the AppleTalk zone Twiddledee:

```
appletalk virtual-net 3 Twiddledee
interface async 1
 encapsulation ppp
 appletalk client-mode
```

# Proxy Network Number Example

Assume that your network topology looks like the one in Figure 12. Also assume that Router A supports only nonextended AppleTalk, that Router B supports only extended AppleTalk (not in transition mode), and that Router C supports only extended AppleTalk.

***Figure 12 Sample Network Topology***



If Router C generates an NBP hookup request for Zone A, Router B will convert this request to a forward request and send it to Router A. Because Router A supports only nonextended AppleTalk, it does not handle the forward request and ignores it. Hence, the NBP lookup from Router C fails.

To work around this problem without putting a transition router adjacent to the nonextended-only router (Router A), you could configure Router D with an NBP proxy.

If you configured Router D with an NBP proxy as follows, any forward requests received for Zone A are converted into lookup requests, and, therefore, the nonextended router for Network 60 can properly respond to NBP hookup requests generated beyond Router C. The following example shows the command needed to describe this configuration:

```
appletalk proxy 60 A
```

# AppleTalk Interenterprise Routing Example

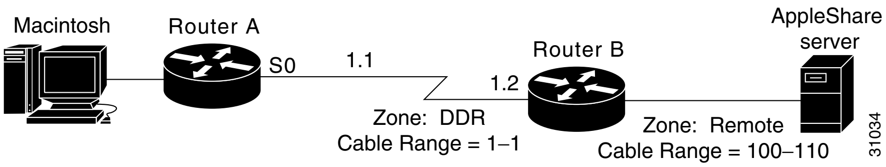
The following example shows how to configure AppleTalk interenterprise routing. It configures domain 1, which is named "France," and places Ethernet interface 2 into this domain.

```
appletalk domain 1 name France
appletalk domain 1 remap-range in 10000-19999
appletalk domain 1 remap-range out 200-299
appletalk domain 1 hop-reduction
!
interface ethernet 2
 no ip address
 no keepalive
 appletalk cable-range 300-300 300.6
 appletalk zone Europe
 appletalk domain-group 1
```

# AppleTalk over DDR Example

The following example describes how to configure AppleTalk to run over a DDR interface, as illustrated in Figure 13. When configuring AppleTalk over DDR, you must specify DDR on the interface on which the static neighbor resides before you specify the static route itself. Also, the Cisco IOS software must know the network address of the static neighbor before you specify the static route. Otherwise, the software will not know to which interface the static neighbor is connected. To open an AppleTalk DDR link, there must be at least one AppleTalk access list bound to a dialer group.

*Figure 13 AppleTalk over DDR Configuration*



The following example shows the steps required to configure AppleTalk over DDR on Router A:

**Step 1** Configure an access list and dialer group.

```
access-list 601 permit cable 100-110
dialer-list 4 list 601
```

**Step 2** Configure the serial interface.

```
interface serial 0
 dialer in-band
 dialer string 1234
 appletalk cable 1-1 1.1
 appletalk zone DDR
 dialer-group 4
 apple distribute-list 601 in
```

**Step 3** Create the static route.

```
appletalk static cable 100-110 to 1.2 zone Remote
```

**Step 4** Open the Chooser on the Macintosh.

**Step 5** Select any AppleTalk service (such as AppleShare, LaserWriter, and so on) in zone Remote to cause Router A to dial up Router B to open a DDR link between them.

**Step 6** Select an AppleTalk file server in the zone Remote. After some time, AppleTalk services appear in zone Remote. Select the one that you need.

**Step 7** Close the Chooser.

**Step 8** Open the AppleTalk session to the remote service.

**Step 9** After the AppleTalk session is finished, close the connection to the remote service. The DDR link should go down after the DDR idle time has elapsed.

Instead of creating a static route in Step 3, you can create a floating static route. The following example adds a floating static route to cable-range 10-11 in the Eng zone with AppleTalk address 6.5 as the next hop router:

```
appletalk static cable-range 10-11 to 6.5 floating zone Eng
```

## AppleTalk Control Protocol for PPP Example

The following example shows how to set up your router to accept AppleTalk client requests on interfaces 1 and 3, using the virtual network number 3 and the AppleTalk zone Twiddledee:

```
Router> enable
Router# config terminal
Router(config)# appletalk virtual-net 3 Twiddledee
Router(config)# interface async 1
Router(config-int)# encapsulation ppp
Router(config-int)# appletalk client-mode
Router(config-int)# interface async 3
Router(config-int)# encapsulation ppp
Router(config-int)# appletalk client-mod
```

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.