Chapter 1
Routing DECnet

1

This chapter describes the Cisco Systems implementation of DECnet Phase IV for the Cisco network server product line. Topics and tasks described in this chapter include:

- Cisco's implementation of DECnet
- An overview of DECnet routing and addressing
- How to enable DECnet routing
- How to configure interarea and intraarea routing costs
- How to configure access lists
- How to set default routers and priority values
- How to fine-tune DECnet performance parameters, including fast switching

DECnet Phase V is equivalent to ISO CLNS, which is described in the chapters "Switching ISO CLNS" and "ISO CLNS Routing Protocols." Support for DECnet Phase IV/Phase V conversion is discussed in this chapter.

Cisco's Implementation of DECnet

Digital Equipment Corporation (DEC) designed the DECnet stack of protocols in the 1970s as part of its Digital Network Architecture (DNA). DECnet has been evolving through its lifetime to its present form known as Phase IV. DECnet support on a Cisco router includes local-area and wide-area DECnet Phase IV routing over Ethernet, Token Ring, FDDI, and serial lines, with the following differences:

- The router uses HDLC framing rather than DEC's DDCMP framing for point-to-point lines. If you construct a network using both Cisco Systems and DEC equipment, you must ensure that each point-to-point line has the same type of equipment on both ends.
- Cisco and DECnet Phase IV routers have incompatible X.25 support. As with point-to-point lines, you must use a single vendor's equipment on the X.25 portion of your network.
- The method of DECnet encapsulation over Token Ring differs between Cisco and DEC equipment. As a result, Cisco and DEC equipment do not interoperate over Token Ring.
- Cisco gives you additional security options through access lists.

Cisco routers can support the Address Translation Gateway (ATG), which allows the router to participate in multiple, independent DECnet networks, and to establish a user-specified address translation table for selected nodes between networks.

DEC uses some nonroutable protocols that are not part of the DECnet stack. Neither Cisco nor DEC routers can route protocols like MOP (discussed later in this chapter), and LAT, the DEC terminal server protocol. These protocols must be bridged; bridging concepts are described in the chapter "Configuring Transparent Bridging" of this manual.

DECnet Phase IV Addresses

DECnet Phase IV addresses are specified by area number and node number separated by a period. DECnet addresses are written as a dotted pair of area and node numbers. For example, **53.6** is node 6 in area 53.

DECnet hosts exist as a *node* (host) in an *area*. Do not confuse the concept of *area* with an area defined by the IP, XNS, or other routing protocols. Unlike these protocols, DECnet allows for an area to span many routers, and for a single cable to have many areas attached to it. Therefore, if a host (such as a router) exists on many cables, it uses the same area/node for itself on all of them. Note how this differs from other routing protocols where each interface is given a different internetwork address. Figure 1-1 shows the DECnet approach.

DECnet hosts do not use manufacturer assigned MAC layer addresses. Instead, network level addresses are embedded in the MAC layer address according to the formula described below.

The area number is six bits long (1 through 63); the node number is 10 bits long (1-1023). To derive a MAC address from a DECnet node number, convert the dotted decimal address into a 16-bit number using the formula: (1024 * area) + node. This 16-bit address is appended to the address AA00.0400 in byte-swapped order, with least significant byte first. The following example illustrates how to convert the DECnet address *12.75*:

- 12 * 1024 + 75 = 12363 (base 10) = 304B (base 16 or hex notation)

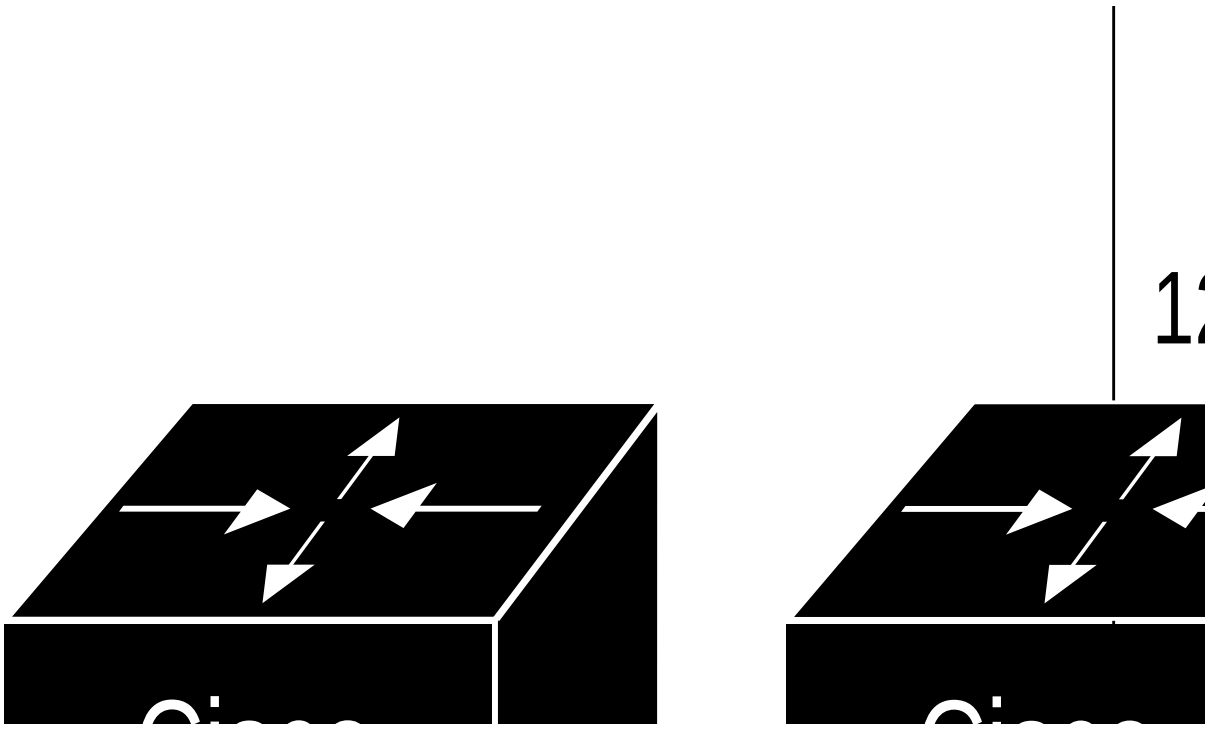
The resulting MAC address is AA00.0400.4B30.

Note: 802.5 Token Ring media transmit bytes in bit-reversed order; the MAC address defined above (for Ethernet) would be 5500.2000.D200 for Token Ring

---

You can also use the EXEC **show interfaces** command to obtain the MAC address once DECnet routing is enabled.

Figure 1-1 DECnet Nodes and Areas



- The DECnet Phase IV protocol associates addresses with machines, not interfaces. Therefore, a router can have only one DECnet Phase IV address for each DECnet network in which it participates. A DECnet MAC-level address is simply an encoded version of the 16-bit area/node combination. This explains why Ethernet interface addresses change on a Cisco router when the DECnet protocol is enabled.
- DECnet does not have the equivalent of the IP Address Resolution Protocol (ARP); DECnet hosts simply advertise their presence with periodic Hello packets. Routers build local routing tables and hosts learn each other's addresses by listening to host Hello messages. Hosts learn about nearby routers by listening to router Hello messages.

You do not have to set each interface address manually; the **decnet routing** global configuration command automatically assigns an address to each interface for which you entered a **decnet cost** configuration command. (These commands are described later in this chapter.)

The parameters in the Cisco Systems implementation of DECnet are a subset of the parameters you can modify in DEC's Network Control Program (NCP). Cisco Systems uses the same names, the same range of allowable values, and the same defaults wherever possible. Note that you must use the configuration commands to set DECnet parameters; the Cisco Systems DECnet implementation does not set parameters by communicating with NCP.

Configuring DECnet Routing

Follow these steps to start configuring your router for DECnet routing:

Step 1: Enable DECnet routing and specify system-wide host addresses; use the **decnet routing** global configuration command.

Step 2: After DECnet routing has been enabled, a cost must be assigned to each interface over which DECnet should run. This enables the interface. DECnet nodes route towards a destination using the lowest path cost, so you should base your cost values on interface throughput. Use the **decnet cost** interface subcommand to set a cost value for an interface.

Step 3: Next, specify with the **decnet node-type** command the node type, either an area router—Level 1 and Level 2—or a local router routing DECnet Phase IV at Level 1 only.

Step 4: You can alter the maximum node number and maximum area number with the optional **decnet max-address** and **decnet max-area** commands.

Step 5: Finally, you must specify several commands for either intraarea or interarea routing. These commands and their parameters must be chosen carefully, as they are dependent on each other's values in many cases.

The following sections take you through these steps in detail, as well as all the optional commands for managing performance, security, Phase IV/V conversion, and so on. The section "DECnet Configuration Examples" shows complete configuration examples for many common situations.

Note: If you plan to use DECnet and Novell routing concurrently on the same interface, you must enable DECnet routing first and then enable Novell routing without specifying the optional MAC address. If Novell routing is enabled before DECnet routing, Novell routing will be disrupted.

Enabling DECnet Routing

To enable or disable DECnet routing, use the **decnet routing** global configuration command:

decnet routing *decnet-address*
no decnet routing

The argument *decnet-address* takes as its value an address in DECnet format X.Y, where X is the area number and Y is the node number. There is no default router address; you must specify this parameter for DECnet operation.

Example:

In the example below, DECnet routing is enabled for the router in area 21 with node number 456:

```
decnet routing 21.456
```

Note: Enabling DECnet changes the MAC addresses of the router's interfaces. This is not a problem on routers equipped with nonvolatile memory. On systems that attempt to get their IP network addresses from network servers rather than from nonvolatile memory, there may be a problem with the hardware addresses changing and confusing other IP-speaking hosts. If you are attempting to use DECnet on such a configuration, be sure to set all global DECnet parameters before enabling DECnet routing on the interfaces.

Assigning the Cost

After DECnet routing has been enabled, you must assign a cost to each interface over which you want DECnet to run. (Assigning a cost in effect enables DECnet routing for an interface.) Most DECnet installations have an individualized routing strategy for using costs. Therefore, check the routing strategy used at your installation to ensure that costs you specify are consistent with those set for other hosts on the network.

The **decnet cost** interface subcommand sets a cost value for an interface:

decnet cost *cost-value*
no decnet cost

The argument *cost-value* is an integer from 1 to 63. There is no default cost for an interface, although a suggested cost for FDDI is 1, for Ethernet is 4 and for serial links is greater than 10. Use the **no decnet cost** subcommand to disable DECnet routing for an interface.

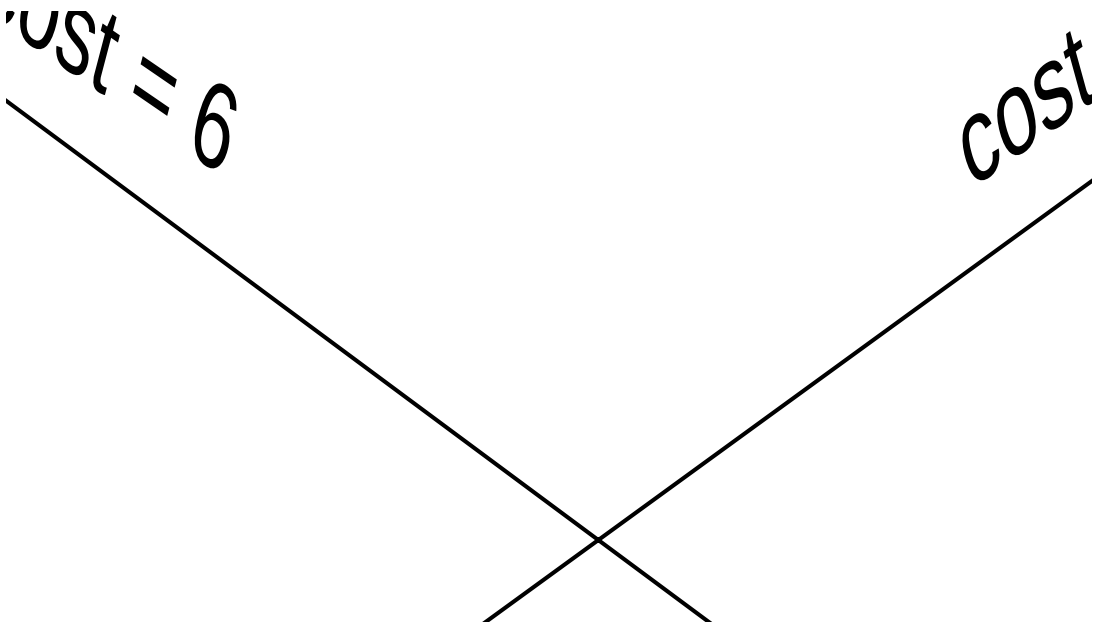
Example:

The example below establishes a DECnet routing process for the router at 21.456, then sets a cost of four for the Ethernet 0 interface.

```
decnet routing 21.456
interface ethernet 0
decnet cost 4
```

Figure 1-2 shows four routers, three Ethernets and the various routes linking them. Each link has a different cost associated with it. The least expensive route from Router 7 to Router 20 is via Router 12.

Figure 1-2 DECnet Cost Values



Specifying the Node Type

Before you use many of the global and interface configuration commands, you must specify the node type with the **decnet node-type** global configuration command. The **decnet node-type** command specifies the node type for the router.

decnet node-type {**routing-iv**|**area**}

The options are either **area** or **routing-iv**. If you specify **area,** the router participates in the DECnet routing protocol with other area routers, as described in the DEC documentation, and routes packets from and to routers in other areas. This is sometimes referred to as Level 2, or interarea, routing. An area router does not just handle interarea routing; it also acts as an intraarea or Level 1 router. If you specify **routing-iv** (the default), the router acts as an intraarea (standard DECnet Phase IV, Level 1 router) and ignores Level 2 routing packets. In this mode, it routes packets destined for other areas via the least-cost path to an interarea router, exchanging packets with other end-nodes and routers in the same area.

Specifying Node Numbers and Area Sizes

DECnet routers do not have the concept of aging out a route. Therefore, all possible areas or nodes must be advertised as unreachable if they cannot be reached. Since it is best to keep routing updates small, you need to indicate the default maximum possible node and area numbers that can exist in the network. The default value for a node address to be given in an update is 1023.

You can use the **decnet max-address** global configuration command to configure the router with a different maximum node address, as follows:

decnet max-address *value*

The argument *value* is a number, less than or equal to 1023, that represents the maximum node address possible on the network. In general, all routers on the network should use the same value for this parameter.

Example:

The example below configures a small network (spanning just a department). The desire is to keep routing updates as small as possible, so the maximum address value is set to 300 instead of the default of 1023.

```
decnet max-address 300
```

Use **decnet max-area** global configuration command to set the largest number of areas that the router can handle in its routing table. The syntax is as follows:

decnet max-area *value*

The argument *value* is an area number from 1 to 63; the default is 63. Like the **decnet max-address** command value, this parameter controls the sizes of internal routing tables and of messages sent to other nodes. All routers on the network should use the same maximum address value.

Example:

In this example, the maximum number of areas that the router will save in its routing table is 45.

```
decnet max-area 45
```

Specifying the Maximum Route Cost for Interarea Routing

The **decnet area-max-cost** global configuration command sets the maximum cost specification value for *interarea* routing. The syntax of this command follows:

decnet area-max-cost *value*

The argument *value* determines the maximum cost for a route to a distant area that the router may consider usable; the router treats as unreachable any route with a cost greater than the value you specify. A valid range for cost is from 1 to 1022; the default is 1022. This parameter is only valid for area routers. Make sure you have used the **decnet node-type area** command before using this command.

Example:

In this example, the node type is specified as area and the maximum cost is set to 500. Any route whose cost exceeds 500 will be considered unreachable by this router.

```
decnet node-type area
decnet area-max-cost 500
```

Use the **decnet area-max-hops** global configuration command to set the maximum hop count value for *interarea* routing as follows:

decnet area-max-hops *value*

The argument *value* determines the maximum number of hops for a usable route to a distant area. The router treats as unreachable any route with a count greater than the value you specify. A valid range for the hop count is from 1 to 30; the default is 30. This parameter is only valid for area routers. Make sure you've used the **decnet node-type area** command before using this command.

Example:

This example sets the node type to area, then sets a maximum hop count of 21. This was done because it is a small network with relatively few routers for interarea routing, so a route with a large hop count is liable to represent a problem, not an efficient route.

```
decnet node-type area
decnet area-max-hops 21
```

Specifying the Maximum Route Cost for Intraarea Routing

The **decnet max-cost** global configuration command sets the maximum cost specification for *intraarea* routing. The router ignores routes within the router's local area that have a cost greater than the corresponding value of this parameter. The syntax for this command follows:

decnet max-cost *value*

The argument *value* is a cost from 1 to 1022 (the default).

Example:

In this example, the node type is specified as DECnet Phase IV and the maximum cost is set to 335. Any route whose cost exceeds 335 will be considered unreachable by this router.

```
decnet node-type routing-iv
decnet max-cost 335
```

Use the **decnet max-hops** global configuration command to set the maximum hop count specification value for *intraarea* routing, as follows:

decnet max-hops *value*

The argument *value* is a hop count from 1 to 30 (the default). The router ignores routes that have a hop count greater than the corresponding value of this parameter.

Example:

This example sets the node type to DECnet Phase IV routing, then sets a maximum hop count of 2.

```
decnet node-type routing-iv
decnet max-hops 2
```

Configuring Maximum Visits

Use the **decnet max-visits** global configuration command to set the limit on the number of times a packet can pass through a router.

decnet max-visits *value*

The *value* keyword can vary from 1 to 63 (the default). If a packet exceeds *value*, the router discards the packet. DEC recommends that the value of the **max-visits** parameter be at least twice that of the **max-hops** parameter, to allow packets to still reach their destinations when routes are changing.

Example:

This example of intraarea routing configuration specifies Phase IV routing, a maximum hop count of 28, and maximum number of visits of 62 (which is more than twice 28).

```
decnet node-type routing-iv
decnet max-hops 28
decnet max-visits 62
```

Configuring Path Selection

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

Limiting the size of the routing table will not affect your router's ability to recover from network failures transparently provided that you do not make the maximum number of paths too small. If more than the specified number of equal cost paths exist and one of those paths suddenly becomes unusable, the router will discover an additional path from the paths it has been ignoring.

The first of the optional path global configuration commands, **decnet max-paths**, defines the maximum number of equal cost paths to a destination that the router will keep in its routing table, with the following syntax:

decnet max-paths *value*

The argument *value* is a decimal number equal to the maximum number of equal cost paths the router will save. The highest value accepted is 31; the default value is 1.

Example:

In the following example, some destinations have six equal cost paths, so the example specifies that the router will save no more than three equal cost paths.

```
decnet max-paths 3
```

The **decnet path-split-mode** global configuration command also helps you make decisions about equal cost paths; it specifies how the router will split the routable packets between equal cost paths. This command has two forms, as shown:

decnet path-split-mode normal
decnet path-split-mode interim

The keyword **normal** selects normal mode (the default), where equal cost paths are selected on a round-robin basis. The keyword **interim** specifies that traffic for any particular (higher-layer) session is always routed over the same path. This mode supports older implementations of DECnet (VMS Versions 4.5 and earlier) that do not support out-of-order packet caching. Other sessions may take another path, thus utilizing equal cost paths that a router may have for a particular destination.

Altering DECnet Defaults

In general, you need not modify the DECnet parameters. However, under special circumstances, or when using a specific configuration, you will see better performance if you alter some of the default parameters. This section will guide you through those special circumstances.

Adjusting Timers and the Route Cache

The router broadcasts Hello messages on all interfaces with DECnet enabled. Other hosts on the network use the Hello messages to identify the hosts with which they can communicate directly. The router sends Hello messages every 15 seconds by default. On extremely slow serial lines, you may want to increase this value to reduce overhead on the line using the **decnet hello-timer** interface subcommand.

decnet hello-timer *value*
no decnet hello-timer

The keyword *value* varies from 1 to 8191 seconds; the default is 15 seconds.

Example:

The following example increases the Hello interval to 2 minutes (120 seconds) on interface serial 1.

```
interface serial 1
decnet hello-timer 120
```

By default, Cisco's DECnet routing software implements fast switching of DECnet datagrams. There are times when it makes sense to disable fast switching. This is especially important when using rates slower than T1.

Fast switching uses memory space interface cards. In situations where a high bandwidth interface is writing large amounts of information to a low bandwidth interface, additional memory could help avoid congestion on the slow interface (also known as big-pipe/little-pipe problems). Use the **no decnet route-cache** interface subcommand to turn off fast switching.

decnet route-cache
no decnet route-cache

In a network where changes occur infrequently or do not need to be responded to immediately (it is small and uncomplicated, applications are not particularly sensitive to delays or occasional packet loss, slow serial links, and so on), increasing the time between routing updates reduces the amount of unnecessary network traffic. The **decnet routing-timer** interface subcommand specifies how often the router sends routing updates that list all the hosts that the router can reach. Other routers use this information to construct local routing tables. DEC calls this parameter the *broadcast routing timer* because they have a different timer for serial lines; the Cisco Systems DECnet implementation does not make this distinction. The syntax for the **decnet routing-timer** interface subcommand follows:

decnet routing-timer *value*
no decnet routing-timer

The argument *value* specifies a time from 1 to 65535 seconds; the default is 40 seconds. The **no decnet routing-timer** command restores this default.

Example:

In the following example, a serial interface is set to broadcast routing updates every two minutes.

```
interface serial 0
decnet routing-timer 120
```

Specifying the Designated Router

The *designated* router is that router to which all end nodes on an Ethernet communicate if they do not know where else to send a packet. The designated router is chosen through an election process in which the router with the highest priority gets the job. When two or more routers on a single Ethernet in a single area share the same highest priority, the unit with the highest node number is elected. You can reset a router's priority to help ensure that it is elected designated router in its area.

Priority may be changed with the **decnet router-priority** interface subcommand, as shown below:

decnet router-priority *value*

The argument *value* can range from zero through 127; the default priority is 64.

Example:

In the following example, interface Ethernet 1 is set to a priority of 110.

```
interface ethernet 1
decnet router-priority 110
```

Configuring DECnet on Token Ring

The Cisco routers support DECnet on Token Ring interfaces.

Note: The method of DECnet encapsulation over Token Ring differs between Cisco and DEC equipment. As a result, Cisco and DEC equipment do not interoperate over Token Ring.

Configuring a Cisco router for DECnet on Token Ring is very similar to configuring DECnet on Ethernet. The only difference is the specification of a Token Ring rather than an Ethernet interface. The syntax for the interface command in shown below:

interface tokenring *number*

The parameter *number* refers to the interface number.

Example:

The example below sets interface Token Ring 0 for DECnet routing.

```
interface tokenring 0
decnet cost 4
```

Managing Traffic Using DECnet Access Lists

There are two forms of DECnet access lists: one that specifies a single address (a standard list) and one that specifies two addresses (an extended list). See the section "Configuring IP Access Lists" in the chapter "Routing IP" for general information about setting up access lists.

Configuring DECnet Access Lists

Use the **access-list** global configuration command to create an access list.

access-list *list* {**permit**|**deny**} *source source-mask*
no access-list *list*

The argument *list* is an integer you choose between 300 and 399 that uniquely identifies the access list. The **permit** and **deny** keywords decide the access control action when a match occurs with the address arguments.

The standard form of the DECnet access list has a DECnet *source* followed by a *source-mask*, also in DECnet address format, with bits set wherever the corresponding bits in the address should be ignored. DECnet addresses are written in the form *area.node* (for example, 50.4 is area 50, node 4). All addresses and masks are in decimal form.

Note: In contrast with IP masks, a DECnet mask specification of "all ones" is entered as the decimal value 1023. In IP, the equivalent is 255.

Example

This example sets up access list 300 to deny packets coming from node 4.51 and permit packets coming from 2.31.

```
access-list 300 deny 4.51 0.0
access-list 300 permit 2.31 0.0
```

Configuring Extended Access Lists

Use this global configuration command to create extended access lists:

access-list *list* {**permit**|**deny**} *source source-mask destination destination-mask*
no access-list *list*

The extended form of the DECnet access list has a source DECnet address and mask pair followed by a destination DECnet address and mask pair.

The argument *list* is an integer you choose between 300 and 399 that uniquely identifies the access list. The **permit** and **deny** keywords decide the access control action when a match happens with the address arguments.

Example:

In the example below, access list 301 is configured to allow traffic from any host in networks 1 and 3. It implies no other traffic will be permitted. (The end of a list contains an implicit "deny all else" statement.)

```
access-list 301 permit 1.0 0.1023 0.0 63.1023
access-list 301 permit 3.0 0.1023 0.0 63.1023
```

DECnet Connect Initiate Filtering

DECnet access lists can be used to filter *connect initiate* packets. This means that you can filter by DECnet object type, such as MAIL.The syntax for the connect initiate filter version of DECnet access lists is:

access-list *list* {**permit**|**deny**} *source source-mask* [*destination destination-mask*] [*connect-entries*]
no access-list *list*

The argument *list* is the access list number in the range 300-399.

The argument pair *source source-mask* is the source address and mask.

The argument pair *destination destination-mask* are the optional destination address and mask.

The *connect-entries* are the optional entries used to match connect packets. These entries are as follows:

{**eq**|**neq**} [*src-object*] [*dst-object*] [*identification*]
eq any

For the **eq**|**neq** option:

- If **eq** is specified, the item matches the packet if all the specified parts of *src-object*, *dst-object*, and *identification* match data in the packet.
- If **neq** is specified, the item matches the packet if *any* of the specified parts do not match the corresponding entry in the packet.

The argument *src-object* consist of two parts:

- The keyword **src**
- The argument *obj-spec*

  The argument *obj-spec* can be one of the following:

  — *relop object-number*.

  — **exp** *regex*.

— **uic** [*group,user*]—In this case the bracket symbols are literal; they must be entered.

The argument [*group, user*] is a a numeric UID expression. The group and user parts may be specified in decimal, octal by prefixing the number with a 0, or hex by prefixing the number with 0x. The **uic** expression is displayed in **show** displays as an octal number.

The argument *relop* can be one of the following relational operator keywords:

— **eq**—equal to

— **neq**—not equal to

— **lt**—less than

— **gt**—greater than

The argument *object-number* is a numeric DECnet object number.

The argument *regex* is a regular expression that matches a string.

Note: Regular expressions are described in the appendix "Pattern Matching."

The argument *dst-object* consist of two parts:

- The keyword **dst**
- The argument *obj-spec* (described above)

The argument *identification* may include any of the following:

- **id** *regex*
- **password** *regex*
- **account** *regex*

The argument *regex* is a regular expression that matches a string.

Table 1-1 lists DECnet object numbers.

Table 1-1 Common DECnet Object Numbers

Name	Number	Description
FAL	17	File Access Listener
HLD	18	Host Loader
NML	19	Network Monitor Link/NICE
MIRROR	25	Loopback mirror
EVL	26	Event logger

Name	Number	Description
MAIL	27	Mail
PHONE	29	Phone
NOTES	33	VAX Notes
CTERM	42	Terminal sessions
DTR	63	DECnet Test Sender/Receiver

Connect Initiate Filter Configuration Considerations

The *obj-spec* may be one of three formats.

- In the first format, you specify a relational operator and an object number. For example:
  - eq 17—Equal to 17 (MAIL)
  - gt 128—Greater than 128

- The second format is a regular expression that matches a string. For example:
  - exp ^SYSTEM$—This expression exactly matches the string *SYSTEM*.
  - exp USER—This expression matches any string containing the substrate *USER*.

- The third format matches a UIC. For example:
  - uic [1,4]

The second and third formats may be used separately, or combined. If specified separately, they might appear as follows:

- src uic [1,4]
- src exp USERNAME

If combined, the *obj-spec* might appear as:

- src exp USERNAME uic [1,4]

The **id**, **password**, and **account** keywords all take regular expressions as arguments and match access information in the packet.

The special format **eq any** matches any connect packet.

Connect Initiate Filtering Examples

The following examples illustrate specification of access lists for connect initiate packet filtering.

Example 1: Match MAIL Object

The following example illustrates an access list for matching all connect packets for the MAIL object:

```
access-list 300 permit 0.0 63.1023 eq dst eq 27
```

Example 2: Match Connect Packets Except FAL Object

The following example illustrates an access list for matching all connect packets *except* for the FAL object:

```
access-list 300 permit 0.0 63.1023 neq dst eq 17
```

Example 3: Match Connect Packets for Access ID

The following example illustrates an access list for matching all connect packets where the access identification was *SYSTEM*:

```
access-list 300 permit 0.0 63.1023 eq id ^SYSTEM$
```

Example 4: Match Connect Packets from Area 1

The following example illustrates an access list for matching all connect packets from area 1 to the MAIL object where *SYSTEM* is the originating user:

```
access-list 300 permit 1.0 0.1023 eq src exp ^SYSTEM$ dst eq 27
```

Example 5: Match Any Connect Packet

The following example illustrates an access list for matching any connect packet:

```
access-list 300 permit 0.0 63.1023 eq any
```

Note: The configuration specification in Example 5 can be used at the end of a list to permit any packets not already matched.

Configuring Access Groups

The **decnet access-group** interface subcommand applies an access list to an interface.

decnet access-group *list*

The argument *list* can be either a standard or extended DECnet access list. A standard DECnet access list applies to destination addresses in this case.

Example:

The following example applies access list 389 to interface Ethernet 1.

```
interface ethernet 1
decnet access-group 389
```

Configuring In- and Out-Routing Filters

The **decnet in-routing-filter** interface subcommand provides access control to Hello messages, or routing information received on this interface. Addresses that fail this test are treated as unreachable. The full syntax of the command follows.

decnet in-routing-filter *list*
no decnet in-routing-filter

The argument *list* is a standard DECnet access list.

The **no decnet in-routing-filter** command removes access control.

Example:

In the following example, interface Ethernet 0 is set up with a DECnet in-routing filter of 321, which means that any Hello messages sent from addresses that are denied in list 321 will be ignored. Additionally, all node addresses listed in received routing messages on this interface will be checked against the access list, and only routes passing the filter will be considered usable.

```
interface ethernet 0
decnet in-routing-filter 321
```

The **decnet out-routing-filter** interface subcommand provides access control to routing information being sent out on this interface. Addresses that fail this test are shown in the update message as unreachable.

decnet out-routing-filter *list*
no decnet out-routing-filter

The argument *list* is a standard DECnet access list.

The **no decnet out-routing-filter** command removes access control.

Example:

In the following example, interface Ethernet 1 is set up with a DECnet out-routing filter of 351. This filter is applied to addresses in the transmitted routing updates. Transmitted Hello messages are not filtered.

```
interface ethernet 1
decnet out-routing-filter 351
```

DECnet Phase IV to Phase V Conversion

DECnet Phase V is OSI-compatible and conforms to the ISO 8473 (CLNP/CLNS) and ISO 9542 (ES-IS) standards. See the "Switching ISO CLNS" and "ISO CLNS Routing Protocols" chapters for an explanation of configuring OSI CLNP routing and for a review of the terminology.

DEC has defined an algorithm for mapping a subset of the Phase V address space onto the Phase IV address space, and also an algorithm for converting Phase IV and Phase V packets back and forth. This allows a network administrator to support both Phase IV hosts in Phase V networks, and Phase V hosts in Phase IV networks.

The algorithms defined by DEC perform the following tasks:

- Conversion between Phase IV and Phase V addresses
- Conversion between Phase V and Phase IV addresses
- Advertisement of Phase IV reachability in a Phase V network
- Advertisement of Phase V reachability in a Phase IV network
- Determination of when to perform the packet conversion

Note: Refer to the chapter "Switching ISO CLNS" for details about DECnet Phase V cluster alias support.

Cisco's implementation is identical to DEC's in the conversion algorithms listed above. Cisco also handles cluster alias in the same way.

Cisco's implementation differs from DEC's implementation in how reachability information is advertised. Cisco's implementation allows you to add Phase V support without modifying your existing Phase IV support. Cisco's implementation delays converting packets from Phase IV to Phase V while DEC's implementation converts as soon as possible.

Cisco routers interoperate with DEC routers, and DEC hosts do not differentiate between a Cisco router and a DEC router.

To enable DECnet conversion, you must configure both DECnet and ISO CLNS on your router. In addition, you must turn on conversion with the **decnet conversion** global configuration command. The command syntax is:

decnet conversion *NSAP-prefix*
no decnet conversion

The argument *NSAP-prefix* defines the value used for the IDP when constructing NSAPs from a Phase IV address.

The command **no decnet conversion** disables Phase IV to Phase V conversion for the router.

Example:

To enable DECnet conversion on a Cisco router with the area tag foo and Phase IV address 20.401 using an ISO IGRP router, enter the following configuration commands:

```
clns routing
decnet routing 20.401
decnet max-address 600
!
router iso-igrp foo
net 47.0004.004d.0014.aa00.0400.9151.00
!
decnet conversion 47.0004.004d
!
interface ethernet 0
decnet cost 4
clns router iso-igrp foo
```

It is essential that the area specified in the **decnet routing** command be the same as the local area specified on the **net** command.

Note: The **decnet routing** command is specified with a decimal address, while the **net** command address is specified in hex. In addition, the *NSAP-prefix* specified on the **decnet conversion** command must match one of the NETs for this router.

Designing a Network to Support Both Phase IV and Phase V

DEC Phase V hosts can use either Phase IV or Phase V packet format. A DEC Phase V host chooses the format to use based on the type of router Hello packets that it sees.

Cisco routers with conversion enabled advertise reachability to both Phase IV hosts and Phase V hosts in both Phase IV and Phase V routing updates.

Cisco routers always attempt to deliver packets in their native format *first*. However, if a Phase IV packet arrives at a router with conversion turned on and the router does not have a Phase IV path to the destination address, the router will convert the Phase IV address to Phase V and look in the Phase V routing table. If a path is found there, the packet will be converted to Phase V and delivered.

If a Phase V packet arrives at a router with conversion turned on and the router does not have a Phase V path to the destination address, the router will convert the Phase V address to Phase IV and look in the Phase IV routing table. If a path is found there, the packets will be converted to Phase IV and delivered.

In addition, all packets to a Phase IV host will be delivered in Phase IV format.

The following guidelines should help you design a network that simultaneously supports DECnet Phase IV and Phase V:

- Host connectivity across multiple areas is only possible if a Level 2 path exists for which every Level 2 router in the path supports a common protocol: Phase IV or Phase V. If not all routers support both protocols, those routers which do *must* have conversion enabled.
- Host connectivity across a single area is only possible if a Level 1 path exists for which every Level 1 router in the path supports a common protocol: Phase IV or Phase V. If not all routers support both protocols, those routers which do must *have* conversion enabled.
- The Level 2 backbone *must* have conversion enabled in all Level 2 routers which support an area that needs conversion.

DECnet Configuration Examples

This section includes configuration examples, showing many common DECnet configuration activities.

Establishing Routing; Setting Interfaces; Maximum Address Space

The configuration subcommands in the example below establish DECnet routing on a Cisco router. The first line establishes DECnet routing for a specific address. The second line sets the maximum address space at 1023 addresses. The second section sets a cost of four for the Ethernet 0 interface. The third section sets a cost of ten for the serial 1 interface.

Example:

```
decnet routing 4.27
decnet max-address 1023
interface ethernet 0
decnet cost 4
interface serial 1
decnet cost 10
```

Level 1 and Level 2 Routing; Designated Router

In the first part of this configuration, the router is being set up with an area and node address in the first line, then it is being designated a Level 2 (area) router. In the lines that follow, the two Ethernet interfaces are given costs of four.

Example 1:

```
decnet routing 6.10
decnet node area
!
interface ethernet 0
decnet cost 4
interface ethernet 1
```

```
decnet cost 4
```

To ensure that a specific router is elected the designated router, assign it the highest possible net address and give it a high router priority as shown below.

Example 2:

```
decnet routing 6.1023
decnet node area
!
interface ethernet 0
decnet cost 4
decnet router-priority 127
!
interface serial 0
decnet cost 20
```

In the third example, the router is a Level 1 router in area 7. The serial link is slower than in the previous example (9.6 vs. 56 Kbps) so it has a higher cost.

Example 3:

```
decnet routing 7.12
decnet node routing-iv
!
interface ethernet 0
decnet cost 4
interface ethernet 1
decnet cost 4
interface serial 0
decnet cost 25
```

Phase IV to Phase V Conversion

This example begins by enabling DECnet routing with a specific address of *54.6*. It then specifies the area with the name *Field* (as in Field Offices) with the **router iso-igrp** command. Specification of the ISO IGRP routing process is followed by specification of the **net** command, which assigns an address to the routing process.

At this point you have set the stage for the **decnet conversion** command, which specifies the NSAP prefix address to be used when converting Phase IV addresses to Phase V.

After you have enabled the conversion, you need to name the specific interfaces that you want to route DECnet packets. In this example, the interface Ethernet 0 is enabled, with a cost of ten. The **clns router iso-igrp** command with the *Field* area is needed to specify that the interface will be using ISO-IGRP and Phase V CLNS and that it is part of area *Field*.

You could follow this interface specification with other interface specifications such as Ethernet 1, serial 0, and so on, with the same three commands. You could also go on to specify access lists and other special commands for these specific interfaces.

Example:

```
decnet routing 54.6
clns routing
router iso-igrp Field
net 47.0006.0200.0000.0000.0100.0036.AA00.0400.06D8.00
decnet conversion 47.0006.0200.0000.0000.0100
interface ethernet 0
decnet cost 10
clns router iso-igrp Field
```

Note: Make sure that the area you specify in the **decnet conversion** command (54) is the same as the area you specified for the CLNS network (36). Also note that the DECnet area is specified in *decimal*, and the CLNS area is specified in *hexadecimal*.

The Address Translation Gateway

The Address Translation Gateway (ATG) allows a Cisco router to route traffic for multiple independent DECnet networks and to establish a user-specified address translation for selected nodes between networks. This allows connectivity between DECnet networks that might be otherwise not connectable due to address conflicts between the networks. The ATG allows you to define multiple DECnet networks and map between them. This may be done over all media types.

ATG Command Syntax

The ATG configuration commands are basically a modification to the standard DECnet global configuration commands.

The general syntax of the DECnet ATG command follows:

decnet *network-number keywords*

The argument *network-number* specifies the network number in the range 0 through 3, and the argument *keywords* is one of the configuration keywords (**area-max-cost**, for example). Commands without the *network-number* modifier apply to *network 0*.

You may also establish a translation entry to translate a virtual DECnet address to a real DECnet address by using this global configuration command:

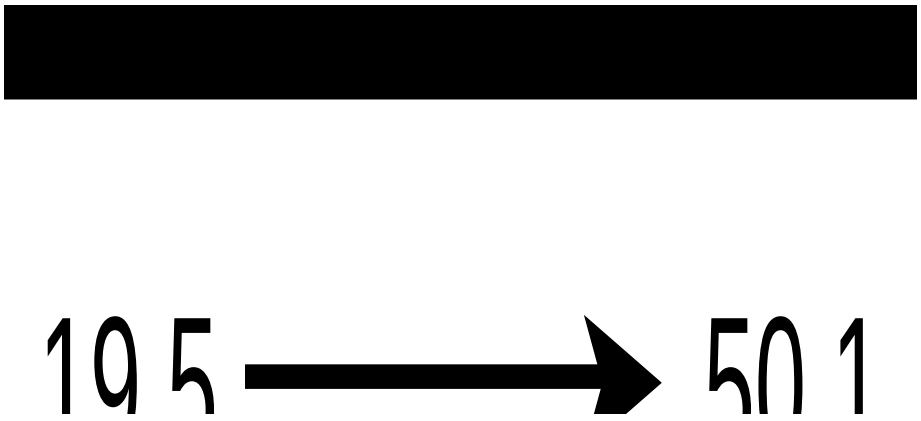
decnet *first-network* **map** *virtual-address second-network real-address*

The arguments *first-network* and *second-network* are DECnet network numbers in the range 0 through 3. The arguments *virtual-address* and *real-address* are specified as numeric DECnet addresses (10.5, for example).

ATG Configuration Examples

In Figure 1-3, the Cisco router is connected to two DECnet networks using Ethernet. The examples following Figure 1-3 refer to the configuration in the figure.

Figure 1-3 ATG Configuration Example



Example:

In Network 0, the router is configured at address 19.4 and is a Level 1 router. In Network 1, the router is configured at address 50.5 and is an area router. At this point, no routing information is exchanged between the two networks. Each network in the router has a separate routing table.

```
decnet 0 routing 19.4
decnet 0 node routing-iv
interface ethernet 0
decnet 0 cost 1
!
decnet 1 routing 50.5
decnet 1 node area
interface ethernet 1
decnet 1 cost 1
```

To establish a translation map, enter these commands:

```
decnet 0 map 19.5 1 50.1
decnet 0 map 19.6 1 19.1
decnet 1 map 47.1 0 19.1
decnet 1 map 47.2 0 19.3
```

Packets in Network 0 sent to address 19.5 will be routed to Network 1 and the destination address will be translated to 50.1. Similarly, packets sent to address 19.6 in Network 0 will be routed to Network 1 as 19.1; packets sent to address 47.1 in Network 1 will be routed to Network 0 as 19.1; and packets sent to *47.2* in Network 1 will be sent to Network 0 as 19.3.

The following table depicts a packet exchange between nodes A and D:

Source			Destination	
A packet addressed as:	19.1	→	19.5	received on Ethernet0
Translates to:	47.1	→	50.1	and is transmitted out Ethernet1
A reply packet:	50.1	→	47.1	received on Ethernet1
Translates to:	19.5	→	19.1	and is transmitted on Ethernet0

Network 0 uses a block of addresses from its area to map the remote nodes. In network 0, the router will advertise nodes 19.5 and 19.6. These nodes must not already exist in network 0.

Network 1 uses another area for the address translation. Since the router will be advertising the availability of area 47, that area should not already exist in network 1 because DECnet area fragmentation could occur.

Only nodes that exist in the maps on both networks will be able to communicate directly. Network 0 node 19.1 will be able to communicate with Network 1 node 50.1 (as 19.5), but will not be able to communicate directly with Network 1 node 60.1.

When naming nodes, use the appropriate address in each network. See the following lists for examples.

Network 0 VMS NCP Command File Sample

```
$ MCR NCP
define node 19.1 name A
define node 19.2 name B
define node 19.3 name C
define node 19.4 name GS
define node 19.5 name D
define node 19.6 name F
```

Network 1 VMS NCP Command File Sample

```
$ MCR NCP
define node 50.1 name D
define node 50.5 name GS
define node 60.1 name E
define node 19.1 name F
define node 47.1 name A
define node 47.2 name C
```

As an additional feature and security caution, DECnet Poor Man's Routing may be used between nodes outside of the translation map as long as those nodes have access to nodes that are in the map, so that a user on node B could issue the following VMS command:

```
$ dir A::D::E::
```

When a Poor Man's Routing connection is made between two networks, only the two adjacent nodes between the networks will have any direct knowledge about the other network. Application-level network access may then be specified to route through the connection.

Note: Cisco does not support Poor Man's Routing directly; the intermediate nodes must be VMS systems with Poor Man's Routing enabled in FAL.'

Limitations of the ATG

Keep the following limitations in mind when configuring the Address Translation Gateway:

- Both nodes that wish to communicate across the ATG must exist in the translation map. Other nodes outside of the map will see route advertisements for the mapped address, but will not be able to communicate with them. An unmapped node trying to communicate with a mapped node will always get the message "Node unreachable." This can be confusing if another nearby node can communicate with mapped nodes because it is also a mapped node.
- Managing a large map can be tedious. Configuration errors will likely cause unpredictable network behavior.
- Third-party DECnet applications could fail if they pass node number information in a data stream (most likely a sign of a poorly designed application).
- Routing information for mapped addresses is static, and does not reflect the reachability of the actual node in the destination network.

DECnet Monitoring Commands

Use the EXEC commands described in this section to obtain displays of activity on the DECnet network.

Displaying DECnet Status

Use the **show decnet interface** command to display the DECnet status and configuration for all interfaces. Enter this command at the EXEC prompt:

show decnet interface [*interface unit*]

When the optional arguments *interface* and *unit* are specified, the relevant information for that particular interface are displayed.

In the following sample output, no specific interface was named, so you see information on all interfaces.

```
Global DECnet parameters for network 0:
  Local address is 19.15, node type is area
  Maximum node is 350, maximum area is 63, maximum visits is 63
  Maximum paths is 1, path split mode is normal
  Local maximum cost is 1022, maximum hops is 30
  Area maximum cost is 1022, maximum hops is 30
Ethernet 0 is up, line protocol is up
  Interface cost is 2, priority is 126, DECnet network: 0
  We are the designated router
  Sending HELLOs every 15 seconds, routing updates 40 seconds
  Smallest router blocksize seen is 576 bytes
  Routing input list is not set, output list is not set
  Access list is not set
  DECnet fast switching is enabled
Serial 0 is up, line protocol is up
  Interface cost is 5, priority is 126, DECnet network: 0
  Sending HELLOs every 15 seconds, routing updates 40 seconds
  Smallest router blocksize seen is 1498 bytes
  Routing input list is not set, output list is not set
  Access list is not set
  DECnet fast switching is enabled
Ethernet 1 is up, line protocol is up
  DECnet protocol processing disabled
```

Displaying the DECnet Address Mapping Information

Use the **show decnet map** command to display the address mapping information used by the DECnet Address Translation Gateway. Enter this command at the EXEC prompt:

show decnet map

Displaying the DECnet Routing Table

Use the **show decnet route** command to display the DECnet routing table. Enter this command at the EXEC prompt:

show decnet route [*decnet-address*]

The optional argument *decnet-address* is a DECnet address and, when specified, the first hop route to that address is displayed. This command may show several routes for a destination when equal cost paths have been set with the **decnet max-paths** command, and when there is more than one equal cost path to a destination. The currently selected route is indicated by an asterisk in the first column of the output. In interim mode, the selected route will never appear to change.

In the following sample output, a DECnet address name was not specified, so the entire routing table is displayed:

```
  Node       Cost  Hops   Next Hop to Node       Expires  Prio
*(Area)        0     0     (Local) ->19.15
```

```
*19.16          2     1   Ethernet0 ->19.16            44     64    V
*19.17          1     1   Ethernet2 ->19.17            31    125    VA
 19.17          2     1   Ethernet0 ->19.17            31    125    VA
*19.22          2     1   Ethernet0 ->19.22            41
```

In the displays:

- The Expires field displays how many seconds from now this entry expires.
- The Prio field is the router priority of this node.
- The V indicates that this is an adjacent Level 1 router; VA or A indicates that this is an adjacent Level 2 (area) router.
- An area node exists on the same local (0 hops) cable.

Displaying DECnet Traffic Statistics

The **show decnet traffic** command shows the DECnet traffic statistics, including datagrams sent, received, and forwarded. Enter this command at the EXEC prompt:

show decnet traffic

Following is sample output:

```
Total: 92275748 received, 758 format errors, 0 unimplemented
       0 not a gateway, 0 no memory, 689 no routing vector
HELLOs: 13113448 received, 26 bad, 15042 other area, 1842481 sent
Level 1 routing: 3919281 received, 0 bad, 580109 other area, 1485567
sent
Level 2 routing: 794130 received, 0 not primary router, 1140858 sent
Data: 73868022 received, 0 not long format, 68 too many visits
      73852256 forwarded, 0 mapped, 10880 returned, 0 converted
      0 access control failed, 10880 no route, 0 encapsulation failed
      0 inactive network, 0 incomplete map
```

In the displays:

- Total: displays the totals of packet types received.
  - The received field is the total of all types of DECnet packets received.
  - The format errors field lists the number of packet that appeared to be DECnet, but were formatted incorrectly. The number in the received field includes these packets.
  - The unimplemented field reports the number of incoming packets that are DECnet control packets, and how many specify a service that the router does not implement, including services implemented to forward Level 1 and Level 2 routing information, and router and end-system Hello packets.
  - The field labeled not a gateway reports the total number of packets received while not routing DECnet.
  - The field labeled no memory is a catch-all that records transaction attempts when the system has run out of memory.

— The field labeled no routing vector indicates that either a routing update came in from another router when the router did not have an adjacency for it, or it had no routing vector for the type of routing update. Execute the **debug decnet-routing** command (in the section "Debugging DECnet") to display additional information.

- HELLOs:displays the number of Hello messages received and sent.
  - — The received field displays the total number of Hello messages received. All protocol types are included.
  - — The `bad` field displays the total number of "bad" Hello messages received. Invoke the EXEC command **debug decnet** to display more information about why the Hello message was judged as bad.
  - — The other area field displays the total number of Hello messages received from nodes on other areas when the router is a Level 1 router only.
  - — The sent field displays the total number of Hello messages sent.

- Level 1 routing:displays the Level 1 routing updates received and sent.
  - — The received field displays the total number of Level 1 routing updates received.
  - — The bad field displays the total number of Level 1 updates received that were judged to be bad.
  - — The other area field displays the total number of Level 1 updates from nodes in other areas.
  - — The sent field displays the total number of Level 1 updates sent.

- Level 2 routing:displays the Level 2 routing updates received and sent.
  - — The received field displays the total number of Level 2 updates received.
  - — The field labeled not primary router should always be zero.
  - — The sent field displays the total number of Level 2 updates sent.

- Data:displays the number of data packets received and sent.
  - — The received field displays the total number of noncontrol (data) packets received.
  - — The field labeled not long format displays the number of packets received which are not in the long DECnet format. This number should always be zero. If it is not, investigate the source of the improperly formatted packets.
  - — The field labeled too many visits lists the number of packets received which have visited too many routers and have been flushed.
  - — The forwarded field lists the total number of packets forwarded.
  - — The mapped field displays the total number of ATG packets mapped.
  - — The returned field lists the total number of packets returned to the sender at the senders' request.

— The converted field displays the number of Phase IV packets converted to Phase V packets.

— The field labeled access control failed lists the packets dropped because access control required it.

— The no route field lists the total packets dropped because the router did not know where to forward them.

— The field labeled encapsulation failed lists the number of packets that could not be encapsulated. This usually happens when there are entries missing in a map for a public data network, such as X.25 or Frame Relay. This can also occur if an interface is set for an encapsulation for which there is no defined DECnet encapsulation (such as PPP on serial interfaces).

— The field labeled inactive network displays the number of packets that appear to come from an unknown interface, or that ATG returned because they did not make sense.

— The field labeled incomplete map counts the number of packets that failed address translation. This usually means a node that is not in the ATG map is trying to access a node in another network advertised by the ATG.

Debugging DECnet

Use the EXEC commands described in this section to troubleshoot and monitor the DECnet network transactions. For each **debug** command, there is a corresponding **undebug** command that turns the message logging off. Generally, you enter these commands with Cisco customer engineers during troubleshooting sessions.

debug decnet-connects

The **debug decnet-connects** command enables logging of all connect packets that are filtered (permitted or denied) by DECnet access lists.

When using connect packet filtering, it may be useful to start with the following basic access list:

```
access-list 300 permit 0.0 63.1023
access-list 300 permit 0.0 63.1023 eq any
```

This allows you to log all connect packets transmitted on interfaces to which you add this list with the **access-group** configuration command. This will allow you to determine those elements on which your connect packets must be filtered.

Consider the following **debug decnet-connect** display:

```
DNET: list 300 item #2 matched src=19.403 dst=19.309 on Ethernet0: per-
mitted

      srcname="RICK" srcuic=[0,017]
      dstobj=42 ID="USER"
```

Here a packet matched the second item in access list 300. The source DECnet address was 19.403, the destination address was 19.309. The packet was permitted and transmitted on interface Ethernet0. The packet had a source object string *RICK* and UIC [0,017]. The destination was object 42. The user specified an ID of *USER*.

Note: Packet password and account information is not logged in the **debug decnet-connects** message, nor is it displayed by the **show access** EXEC command. If you specify **password** or **account** information in your access list, these will be viewable by anyone with access to your router's configuration.

---

debug decnet-packets

The **debug decnet-packets** command enables logging of all DECnet routing updates and Hello packets.

debug decnet-routing

The **debug decnet-routing** command enables logging of all changes made to the DECnet routing table, that is, new routes, routes that change cost, routes that expire, and so on.

DECnet Global Configuration Command Summary

This section provides an alphabetically arranged summary of all the DECnet global interface commands. These commands may appear any place in the configuration file.

[**no**] **access-list** *list* {**permit**|**deny**} *destination destination-mask*

Creates an access. The argument *list* is an integer you choose between 300 and 399 that uniquely identifies the access list. The **permit** and **deny** keywords decide the access control action when a match happens with the address arguments. The argument pair *destination destination-mask* are the optional destination address and mask. The **no** form of the command deletes access lists.

[**no**] **access-list** *list* {**permit**|**deny**} *source source-mask destination destination-mask*

Creates an extended access lists. The extended form of the DECnet access list has a source DECnet address and mask pair followed by a destination DECnet address and mask pair. The argument *list* is an integer you choose between 300 and 399 that uniquely identifies the access list. The **permit** and **deny** keywords decide the access control action when a match happens with the address arguments. The **no** form of the command deletes access lists.

[**no**] **access-list** *list* {**permit**|**deny**} *source source-mask* [*destination destination-mask*] [*connect-entries*]

DECnet access lists can be used to filter connect initiate packets. The argument *list* is the access list number in the range 300-399. The argument pair *source source-mask* is the source address and mask The argument pair *destination destination-mask* are the optional destination address and mask. The *connect-entries* are the optional entries used to match connect packets. The **no** form of the command deletes access lists.

decnet area-max-cost *value*

Sets the maximum cost specification value for *interarea* routing. The argument *value* determines the maximum cost for a route to a distant area that the router may consider usable; the router treats as unreachable any route with a cost greater than the value you specify. A valid range for cost is from 1 to 1022; the default is 1022. This parameter is only valid for area routes.

decnet area-max-hops *value*

Sets the maximum hop count specification value for *interarea* routing. The argument *value* determines the maximum number of hops for a route to a distant area that the router may consider usable; the router treats as unreachable any route with a count greater than the value you specify. A valid range for the hop count is from 1 to 30; the default is 30. This parameter is only valid for area routes.

[**no**] **decnet conversion** *NSAP-prefix*

Enables DECnet conversion. The argument NSAP-prefix defines the value used for the IDP when constructing NSAPs from a Phase IV address. The command **no decnet conversion** disables Phase IV/V conversion on the router.

decnet *network-number keywords*

Specifies ATG. The argument *network-number* specifies the network number in the range 0 through 3, and the argument *keywords* is one of the configuration keywords. Commands without the *network-number* modifier apply to “network 0.”

decnet *first-network* **map** *virtual-address second-network real-address*

Establishes a translation entry to translate a virtual DECnet address to a real DECnet address. The arguments *first-network* and *second-network* are DECnet network numbers in the range zero through three. The arguments *virtual-address* and *real-address* are specified as numeric DECnet addresses.

decnet max-address *value*

Determines the largest node number specification allowed in the current area. The argument *value* is a node number from 1 to 1023; the default is 1023. This parameter controls the sizes of internal routing tables and of messages sent to other nodes.

decnet max-area *value*

Sets the largest area number specification that the router can handle. The max-area keyword takes as its value an area number from 1 to 63; the default is 63.

decnet max-cost *value*

Sets the maximum cost specification for *intraarea* routing. The router ignores routes within the local area that have a cost greater than the corresponding value of this parameter. The argument *value* is a cost from 1 to 1022; the default is 1022.

decnet max-hops *value*

Sets the maximum hop count specification value for *intraarea* routing. The router ignores routes within the local area that have a hop count greater than the corresponding value of this parameter. The argument *value* is a hop count from 1 to 30; the default is 30.

decnet max-paths *value*

Defines the maximum number of equal cost paths to a destination that may be kept by the router. The argument *value* specifies the maximum number of equal cost paths, which is limited to 31. The default value is one, which specifies no multiple paths.

decnet max-visits *value*

Sets the limit on the number of times a packet can pass through a router. The argument *value* is a number from 1 to 63; the default value is 63.

decnet node-type {**area**|**routing-iv**}

Specifies the node type for the router. This command takes another keyword, **area** or **routing-iv,** as its value. If you specify **area,** the router exchanges traffic directly with routers in other areas, and participates in the interarea (Level 2) routing protocol, as well as acting as a intraarea (Level 1) router for its local area. If you specify **routing-iv** (the default), the router acts as an intraarea router, and routes packets out of the area by taking the least cost path to an interarea router.

decnet path-split-mode {**normal**|**interim**}

Sets the mode for splitting the routes between equal cost paths. The keyword **normal** selects the normal mode, where equal cost paths are selected on a round-robin basis. The normal mode is the default. The keyword **interim** selects an interim mode, where traffic for any particular higher level session is always routed over the same path. This mode supports older implementations of DECnet (VMS Versions 4.5 and earlier) that do not support out-of-order packet caching.

[**no**] **decnet routing** *decnet-address*

Enables or disables DECnet routing. The argument *decnet-address* takes as its value an address in DECnet format X.Y, where X is the area number and Y is the node number. There is no default router address; you must specify this parameter for DECnet operation.

DECnet Interface Subcommand Summary

This section provides an alphabetically arranged summary of the DECnet interface subcommands. These commands follow an **interface** command.

[**no**] **decnet access-group** *list*

Applies or removes an access list. The argument *list* can be either a standard or extended DECnet access list. A standard DECnet access list applies to destination addresses in this case.

[**no**] **decnet cost** *cost-value*

Sets or removes a cost value for an interface. The argument *cost-value* is an integer from 1 to 63. There is no default cost for an interface, although a suggested cost for Ethernet is 4, and all hosts on the same cable must share the same value. Use the **no decnet cost** subcommand to disable DECnet routing for an interface.

[**no**] **decnet hello-timer** *value*

Specifies how often the router sends Hello messages. This keyword takes as its value a time from 1 to 8.191 seconds; the default is 15 seconds. The **no** form of the command restores the default.

[**no**] **decnet in-routing filter** *list*

Provides access control to Hello messages or routing information received on this interface. Addresses that fail this test are treated as unreachable. The argument *list* is a standard DECnet access list. The **no** form of the command removes access control.

[**no**] **decnet out-routing-filter** *list*

Provides access control to routing information being sent out on this interface. Addresses that fail this test are shown in the update message as unreachable. The argument *list* is a standard DECnet access list. The **no** form of the command removes access control.

[**no**] **decnet route-cache**

Fast switching and the route cache are normally enabled. If you want to disable fast switching, use the **no** form of the command.

[**no**] **decnet router-priority** *value*

Sets a priority value for use in determining the default router. Argument *value* is a number from 0 to 127; the default is 64. The **no** form of the command restores the default.

[**no**] **decnet routing-timer** *value*

Specifies how often the router sends routing messages. The argument *value* is a time from 1 to 65535 seconds; the default is 40 seconds. The **no** form of the command restores the default.