

Route Maps and Other Objects for Route Tuning

The various routing protocols let you fine-tune activities such as route distribution and aggregation. For some tuning features, you use route maps or other objects to identify the routes that should be subject to your tuning policy. Route maps have the additional ability to set options on matching routes, so that you can make changes to the route that the next hop router can use to apply custom behavior.

Whether you need to create any of these objects is based on what your needs are for fine-tuning the behavior of the routing protocols that you implement. By first evaluating your requirements, you will determine which types of object you need for the tuning command you want to configure.

- Configure Route Maps, on page 1
- Configure Access Lists, on page 6
- Configure AS Path Access Lists, on page 10
- Configure Community Lists, on page 11
- Configure Policy Lists, on page 13
- Configure Prefix Lists, on page 14

Configure Route Maps

You can use route maps for a variety of purposes, with some routing protocols supporting more uses than others. The most typical use is to fine-tune route redistribution into another routing protocol.

Route Map Permit and Deny Clauses

A route map consists of one or more **permit** or **deny** clauses. The sequence of these clauses matter: routes are evaluated against the map top-down, first match wins. If a route does not match any clause, it is considered to not match the route map.

Each permit clause can contain zero or more **match** and **set** statements. The **match** statement determines which routes match the clause, whereas the **set** statements modify some characteristic of the routes, such as the route metric. You do not need any set statements: you can match a route for redistribution (or another service) without changing the routes in any way.

Each deny clause can contain zero or more match statements. But, because “denied” routes simply do not match the route map, it is pointless to include set clauses, because the set action cannot be applied.

Route Map Match and Set Statements

Each route map clause has two types of values:

- A match value selects routes to which this clause should be applied.
- A set value modifies some attribute of the routes.

For example, for each route that is being redistributed, the router first evaluates the match criteria of a clause in the route map. If the route matches the criteria, then the route is redistributed or rejected as dictated by the permit or deny clause. For matches to permit clauses, some of the route's attributes might be modified by the values from the set commands. If the route does not match the criteria, then this clause is not applicable to the route, and the system proceeds to evaluate the route against the next clause in the route map. Scanning of the route map continues until a clause is found that matches the route or until the end of the route map is reached. If there are no matches, the route is considered to not match the route map (equivalent to a deny action).

For the match and set statements in a single clause:

- Multiple match statements are ANDed. That is, a route must satisfy each statement to match the clause.
- Multiple values within a single match statement are ORed. That is, if a route matches any value within that match statement, it is considered to match the statement as a whole.
- If there are no match statements, all routes match the clause.
- If there are no set statements in a route map permit clause, then the feature (such as redistribution) is applied to the route without modification of the route's current attributes.
- Any set statements in a deny clause are ignored. “Denied” routes simply do not match the route map, so it is pointless to include set clauses, because the set action cannot be applied.
- An empty clause, one with no match or set statements, matches any routes that have not been matched by earlier clauses. For example:
  - An empty permit clause allows the redistribution of the remaining routes without modification.
  - An empty deny clause does not allow the redistribution of the remaining routes. This is the default action if a route map is completely scanned, but no explicit match is found.



Note Route-map set statements behave differently depending on where you apply the route map. When you apply a route map to BGP, **set ip next-hop** or **set ipv6 next-hop** changes the BGP NEXT_HOP attribute for matching routes. When you apply a route map to an interface for policy-based routing (PBR), **set ip next-hop** or **set ipv6 next-hop** sets the forwarding next hop for matching packets.

Configure a Route Map

You can use route maps for a variety of purposes, with some routing protocols supporting more uses than others. The most typical use is to fine-tune route redistribution into another routing protocol.

A route map consists of one or more **permit** or **deny** clauses. The sequence of these clauses matter: routes are evaluated against the map top-down, first match wins. If a route does not match any clause, it is considered to not match the route map.

Each permit clause can contain zero or more **match** and **set** statements. The **match** statement determines which routes match the clause, whereas the **set** statements modify some characteristic of the routes, such as the route metric. You do not need any set statements: you can match a route for redistribution (or another service) without changing the routes in any way.

Each deny clause can contain zero or more match statements. But, because "denied" routes simply do not match the route map, it is pointless to include set clauses, because the set action cannot be applied.

For a detailed explanation of how match and set statements are evaluated, carefully read Route Map Match and Set Statements, on page 2.

Before you begin

You can use a variety of other objects in a route map to define match criteria, such as access lists, AS path access lists, community lists, policy lists, and prefix lists. You must create these objects before you create the route map.

For ACL matching, you can use standard or extended ACLs for IPv4 addresses, but extended ACLs only for IPv6. Because the match clauses are based on IPv4 or IPv6 only, ensure your ACLs have the correct address scheme for the match statements.

Also note that the match and set criteria are different for BGP compared to the other routing protocols. Ensure you select the correct match/set criteria for the routing process that will use the route map.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon () for the object.

To delete an unreferenced object, click the trash can icon () for the object.

Step 4 Select **Route Map** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the route map name in the first line of the CLI template (in the **route-map** command).

Step 6 Create the first clause:

a) Click the *redistribution* variable and select one of the following:

- **permit**—Match. Connections that match this rule are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this rule are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, if you use this route map to define which routes get redistributed, "denied" address spaces are simply not redistributed.

b) Click the **sequence-number** variable and enter the number for the clause, from 1-65535.

This number is relative to the other numbered clauses within the route map. A typical practice is to skip count by 10, that is, 10, 20, 30, to leave room to insert new clauses in the future.

Step 7 Click **Show Disabled** and configure the **match** statements for the clause.

a) Click the + next to the **configure** *clause* command to enable it.

b) Click *clause* and either pick **bgp-match-clause** for BGP route maps, or **match-clause** for all other routing protocols.

c) (BGP route maps.) Configure any combination of the following **match** statements to identify the specific routes you are targeting in this clause. Be sure to click the - icon to disable any commands that you do not configure.

- **match as-path**. Click the variable and select the AS Path objects that define the autonomous system numbers to match.
- **match community**. Click the variable and select the community list objects that define the communities to match.
- **match policy-list**. Click the variable and select the policy list objects that define the match criteria for the clause.
- **match tag**. Click the variable and enter the route tag value to match, from 0-4294967295.

d) (All other routing protocols.) Configure any combination of the following **match** statements to identify the specific routes you are targeting in this clause. Be sure to click the - icon to disable any commands that you do not configure. You might need to click + to enable some of these commands.

- **match interface**. Click the variable and select all of the interfaces in the routes to match.
- **configure match ipv4/ipv6 ip address** *list-type*. Enable the correct command for your IP version. Then, click the *list-type* variable and select whether you want to match the IP address in the route based on **access-list** or **prefix-list**. This will add a **match ipv4/ipv6 address** command, where you can click the variable and select either the access lists or prefix lists that define the IP addresses to match.
- **configure match ipv4/ipv6 ip next-hop** *list-type*. Click the *list-type* variable and select whether you want to match the IP address of the next hop router in the route based on **access-list** or **prefix-list**. This will add a **match ipv4/ipv6 next-hop** command, where you can click the variable and select either the access lists or prefix lists that define the IP addresses to match.
- **configure match ipv4/ipv6 ip route-source** *list-type*. Click the *list-type* variable and select whether you want to match the IP address of the route source in the route based on **access-list** or **prefix-list**. This will add a **match ipv4/ipv6 route-source** command, where you can click the variable and select either the access lists or prefix lists that define the IP addresses to match.
- **match metric**. Click the variable and enter the routing metric to match, from 1-4294967295.
- **match route-type**. (OSPF, EIGRP.) Click the variable and select the route type:
  - **external-1**, **external-2**. OSPF or EIGRP external type-1 or type-2 routes.
  - **internal**. OSPF intra-area and interarea routes or EIGRP internal routes.
  - **local**. Locally generated BGP routes.
  - **nssa-external-1**, **nssa-external-2**. External Not So Stubby Area (NSSA) type-1 or type-2 routes.

Step 8 (Optional, permit clauses only.) For permitted, that is, matched routes, you can configure **set** statements to modify the route attributes. You do not need to modify routes; for example, you can redistribute them unchanged.

a) Click **...** > **Duplicate** to the left of the configure match-clause or **configure bgp-match-clause** command within the permit clause. A new **configure** *clause* command is added at the end of the permit clause.

b) Click *clause* and either pick **bgp-set-clause** or **set-clause**, based on what you picked for the match clause.

c) (BGP route maps.) Configure any combination of the following **set** statements to modify the attributes of matching routes. Be sure to click the - icon to disable any commands that you do not configure.

- **configure set as-path** *options*. Click *options* and select **properties**, which adds the following commands you need to configure. By adding items to the paths, even duplicate AS numbers, you lengthen the path and make the route less likely to be selected as the best route.
  - **set as-path prepend** *as-path*. Click *as-path* and enter up to 10 autonomous system numbers to be added to be beginning of the AS_PATH attribute of the route. The change applies to outbound BGP route maps.
  - **set as-path prepend last-as** *value*. Click *value* and enter how many times the system should prepend the advertising neighbor's autonomous system number to the start of the AS_PATH variable. The change applies to inbound BGP route maps.
  - **set as-path tag**. Converts the tag of a route into an autonomous system path. Applies only when redistributing routes into BGP.
- **set community** *community-number properties*. Click *community-number* and enter the community for the route, from 1-4694967295. Optionally, you can click *properties* and add one of the following:
  - **internet**—Routes with this community are advertised to all peers (internal and external).
  - **no-advertise**—Routes with this community are not advertised to any peer (internal or external).
  - **no-export**—Routes with this community are advertised to only peers in the same autonomous system or to only other sub-autonomous systems within a confederation. These routes are not advertised to external peers.
- **set local-preference**. Click the variable and enter the a preference value for the autonomous system path, from 0-4294967295. Unless you change it in the global BGP options, the default preference for BGP routes is 100. The route with the highest preference number is preferred.
- **set weight**. Click the variable and enter the weight for the route, 0-65535. If the router learns about more than one route to the same destination, the route with the highest weight is preferred.
- **set origin** *options*. The origin of a BGP route is based on the path information of the route in the main IP routing table. You can change this by clicking *options* and select how you want to set the BGP origin code.
  - **igp**. Set the origin to the remote Interior Gateway Protocol (IGP) system.
  - **incomplete**. Set the origin as unknown heritage.
- **configure next-hop ipv4/ipv6** *options*. These are separate commands. Select the IPv4 or IPv6 next-hop option. In a BGP route map, this setting changes the BGP NEXT_HOP attribute for matching routes. In a PBR route map, this setting selects the forwarding next hop for matching packets.

- **specific-ip**. Select this option if you want to explicitly set the IP address of the next hop gateway for this route. The **set ip/ipv6 next-hop ip-address** command is added. Click the variable and enter the IP address of the next hop gateway. You can add multiple IP addresses, separated by a space. If the address of the first gateway is unreachable, the next address is tried, and so forth.
- **user-peer-address**. Select this option if you want to set the next hop gateway as the BGP peer's IP address. If you use this option in an outbound route map of a BGP peer, the next hop of the advertised matching routes will be set to be the peering address of the local router, thus disabling the next hop calculation. No additional configuration is required for this command.

- **set ipv4/ipv6 address** *prefix-list*. These are separate commands. Change the IP address of the route based on the contents of the prefix list that you select.
- **set automatic-tag**. Have the system automatically compute a tag value for the route.

d) (All other routing protocols.) Configure any combination of the following **set** statements to modify the attributes of matching routes. Be sure to click the - icon to disable any commands that you do not configure.

- **set metric**. Click the variable and enter the metric value, from 0-4294967295. This value is not used by EIGRP.
- **set metric-type**. Click the variable and select the type of metric:
  - **type-1**, **type-2**. The type of external route in OSPF. Type-2 is the default.
  - **internal**. Sets the Multi Exit Discriminator (MED) value on prefixes advertised to external BGP (eBGP) neighbors to match the Interior Gateway Protocol (IGP) metric of the next hop of the route. This applies to generated, internal BGP (iBGP)-, and eBGP-derived routes.

Step 9 Add permit/deny clauses to complete the route map.

To add a clause, click **...** > **Duplicate** to the left of a **permit** or **deny** line. A new *redistribution sequence-number* clause is added immediately after the clause for which you click the Duplicate command.

Although route map clauses are evaluated in the order of the sequence number rather than the order the appear in the object, it is easier to edit your object if you insert new clauses in sequential order. You cannot move the clauses within the object.

Note that duplicating a clause simply inserts a new, empty clause, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 10 Click **OK** to save the object.

You can now use the object in a routing process configuration, or in a FlexConfig object, for a feature that requires a route map.

Configure Access Lists

An access list object, also known as an access control list (ACL), selects the traffic to which a service will apply. You use these objects when configuring particular features, such as route maps. Traffic identified as

allowed by the ACL is provided the service, whereas "blocked" traffic is excluded from the service. Excluding traffic from a service does not necessarily mean that it is dropped altogether.

You can configure the following types of ACL:

- Extended—Identifies traffic based on source and destination address and ports. Supports IPv4 and IPv6 addresses.
- Standard—Identifies traffic based on destination address only. Supports IPv4 only.

An ACL is composed of one or more access control entry (ACE), or rule. The order of ACEs is important. When the ACL is evaluated to determine if a packet matches a "permit" ACE, the packet is tested against each ACE in the order in which the entries are listed. After a match is found, no more ACEs are checked. For example, if you want to match 10.100.10.1, but exclude the rest of 10.100.10.0/24, the permit entry for 10.100.10.1 must come before the deny entry for 10.100.10.0/24. In general, place more specific rules at the top of an ACL.

Packets that do not match a permit entry are considered to be denied or excluded from the match.

The following topics explain how to configure ACL objects.

Configure Extended Access Lists

Use extended ACL objects when you want to match traffic based on source and destination addresses, protocol and port, or if the traffic is IPv6.

Before you begin

Create any network or port objects you will need in the ACEs you create in the object.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon ( ) for the object.

To delete an unreferenced object, click the trash can icon ( ) for the object.

Step 4 Select **Extended Access List** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the ACL name in the first line of the CLI template (in the **access list** command).

Step 6 Create the ACE that should be the top rule in the ACL.

Each list of commands contained within a single **configure access list entry** command is essentially one ACE, although when deployed, the system might break out the command into a series of ACEs, especially if you include more than one network object.

a) In the configure access list entry command, click *action* and select one of the following:

- **permit**—Match. Connections that match this ACE are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this ACE are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this ACL to define which routes get redistributed, "denied" address spaces are simply not redistributed.

b) In the **permit/deny network** command, click the variables to select the network objects that define the source IP address and the destination IP address of the connection. You can select multiple objects. To specify "any" address, select the any-ipv4 and any-ipv6 objects.

The object or object group cannot contain an FQDN: the objects must specify IP addresses only. FQDN objects are valid in access control rules only.

c) In the **configure permit/deny port** command, click *options* and select one of the following, which will add the associated permit/deny command to the template:

- **any**—If the port does not matter. That is, you are matching any type of IP traffic.
- **any-source**—If the source TCP/UDP port does not matter, but you want to specify the destination port. Click the *destination-port* variable in the **permit/deny port** command and select the port object.
- **any-destination**—If the destination TCP/UDP port does not matter, but you want to specify the source port. Click the *source-port* variable in the **permit/deny port** command and select the port object.
- **source-destination**—If both the source and destination TCP/UDP ports matter. Click the *source-port* and *destination-port* variables in the **permit/deny port** command and select the port objects.

d) In the **configure logging** command, select **disabled**. Logging applies to ACLs that are used for access control, and you cannot use these objects for access control. Thus, the logging options are ignored no matter what you select.

Step 7 Add ACEs to complete the ACL.

To add an ACE, click **...** > **Duplicate** to the left of the **configure access list entry** line. A new ACE group is added immediately after the ACE for which you click the Duplicate command.

Thus, when you have many ACEs in the object, choose wisely which ACE you "duplicate." You cannot move the ACEs within the object, so if you make a mistake, you need to manually recreate the ACE in the correct location.

Note that duplicating an ACE simply inserts a new, empty ACE, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 8 Click **OK** to save the object.

You can now use the object in a route map object, or in a FlexConfig object, for a feature that requires an extended ACL.

Configure Standard Access Lists

Use standard ACL objects when you want to match traffic based on destination IPv4 address only, and the feature you are configuring supports standard ACLs. Otherwise, use extended ACLs.

Before you begin

Create any network objects you will need in the ACEs you create in the object.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon () for the object.

To delete an unreferenced object, click the trash can icon () for the object.

Step 4 Select **Standard Access List** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the ACL name in the first line of the CLI template (in the **access list** command).

Step 6 Create the ACE that should be the top rule in the ACL.

Each list of commands contained within a single **configure action** command is one ACE.

a) In the **configure action** command, click *action* and select one of the following:

- **permit**—Match. Connections that match this ACE are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this ACE are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this ACL to define which routes get redistributed, "denied" address spaces are simply not redistributed.

b) In the **permit/deny host** command, click the variable to select the network object that defines the destination IP address of the connection. The object can specify a network or host address. You can select one object per **permit/deny host** command; click **...** > **Duplicate** on the command to specify additional addresses, which will become unique ACEs with the same action. To specify "any" address, select the any-ipv4 object.

Step 7 Add ACEs to complete the ACL.

To add an ACE, click **...** > **Duplicate** to the left of the **configure action** line. A new ACE group is added immediately after the ACE for which you click the Duplicate command.

Thus, when you have many ACEs in the object, choose wisely which ACE you "duplicate." You cannot move the ACEs within the object, so if you make a mistake, you need to manually recreate the ACE in the correct location.

Note that duplicating an ACE simply inserts a new, empty ACE, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 8 Click **OK** to save the object.

You can now use the object in a route map object, or in a FlexConfig object, for a feature that requires a standard ACL.

Configure AS Path Access Lists

You can use an AS Path access list to filter BGP neighbor updates based on the autonomous system numbers in the updates. Permitted AS numbers have their updates accepted, whereas denied AS numbers have their updates rejected, that is, they are not added to the routing table.

You can also apply AS path filtering in the outbound direction and filter the updates you send to neighbors.

In addition, you can use AS path objects in route maps for BGP address aggregation,

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon ( ) for the object.

To delete an unreferenced object, click the trash can icon ( ) for the object.

Step 4 Select **ASPath** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. The name must be a number in the 1-500 range. Note that this name is also entered as the AS path access list name in the first line of the CLI template (in the **as-path** command).

Step 6 Configure an AS path entry.

Each entry is contained on a single line starting with the *action* option.

a) Click *action* and select one of the following:

- **permit**—Match. Connections that match this rule are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this rule are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this object to define which routes get redistributed, "denied" address spaces are simply not redistributed.

b) Click *regex* and enter the regular expression that defines the AS numbers that should match this entry.

In its simplest form, the regular expression is just a complete AS path number, and you are either permitting or denying route updates from a single autonomous system.

The AS number can be from 1 to 4294967295 or from 1.0 to 65535.65535. The AS number is a uniquely assigned value that identifies each network on the Internet. The system supports asplain and asdot notation

as defined in RFC 5396. Which notation you need to use depends on whether you enable the **bgp asnotation dot** command in the BGP global settings.

Step 7 Add entries to complete the AS path access list.

To add an entry, click **...** > **Duplicate** to the left of the *action* line. A new entry is added immediately after the entry for which you click the Duplicate command.

Thus, when you have many entries in the object, choose wisely which entry you "duplicate." You cannot move the entries within the object, so if you make a mistake, you need to manually recreate the entry in the correct location. The rules are evaluated top down, first match wins.

Note that duplicating an entry simply inserts a new, empty entry, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 8 Click **OK** to save the object.

You can now use the object in a BGP object, route map object, or in a FlexConfig object, for a feature that requires an AS path access list.

Configure Community Lists

If you enable your BGP process to send community information, you can use community lists as a match clause in route maps to set attributes on matching routes. For example, you could change the route preference for certain communities.

A community is an optional attribute or label that a service provider would attach to advertised routes for a group of destinations that share some common attribute. The specific community numbers would be something your ISP would advertise: you would need to obtain the numbers and their meaning from your ISP, and then choose how you want to handle them using a route map.

Community lists are ordered, and matches are determined in a top-down, first match wins method similar to access and prefix lists.

There are two types of community list:

- Standard—Use a standard list when you want to target specific well-known communities, such as the ones obtained from your service provider.
- Expanded—Use an expanded list when you want to match a set of communities based on regular expression matching.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.

- To edit an object, click the edit icon () for the object.

To delete an unreferenced object, click the trash can icon () for the object.

Step 4 Select **Standard Community List** or **Expanded Community List** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the Community List name in the first line of the CLI template (in the **community-list** command).

Step 6 (Standard List.) Configure a community list entry.

Each entry is contained on a single line starting with the *action* option.

a) Click *action* and select one of the following:

- **permit**—Match. Connections that match this rule are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this rule are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this rule to define which routes get redistributed, "denied" address spaces are simply not redistributed.

b) Click *community-number* and enter up to 10 space-separated communities. Multiple communities on a single rule are ANDed, so a match exists only if all communities are matched in the route.

Enter the community in decimal format (1-4294967295) or AA:NN format (each value is 1-66535) based on which numbering method is enabled for your BGP process. Get these numbers from your ISP or other BGP neighbors.

c) (Optional.) Click *properties* and add other well-known communities to the rule.

- **internet**—Routes with this community are advertised to all peers (internal and external).
- **no-advertise**—Routes with this community are not advertised to any peer (internal or external).
- **no-export**—Routes with this community are advertised to only peers in the same autonomous system or to only other sub-autonomous systems within a confederation. These routes are not advertised to external peers.

Step 7 (Expanded List.) Configure a community list entry.

a) Click *action* and select **permit** or **deny**. These actions are explained above.

b) Click *regex* and enter the regular expression that defines the communities that should match this entry.

The order for matching using the * or + character is longest construct first. Nested constructs are matched from the outside in. Concatenated constructs are matched beginning at the left side. If a regular expression can match two different parts of an input string, it will match the earliest part first. For more information about writing regular expressions, see the "Regular Expressions" appendix of the Cisco IOS Terminal Services Configuration Guide.

Step 8 Add entries to complete the community list.

To add an entry, click **...** > **Duplicate** to the left of the *action* line. A new entry is added immediately after the entry for which you click the Duplicate command.

Thus, when you have many entries in the object, choose wisely which entry you "duplicate." You cannot move the entries within the object, so if you make a mistake, you need to manually recreate the entry in the correct location.

Note that duplicating an entry simply inserts a new, empty entry, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 9 Click **OK** to save the object.

You can now use the object in a route map or routing process, or in a FlexConfig object, for a feature that requires a community list.

Configure Policy Lists

You can use policy lists in route maps as a replacement for one or more match clauses. Thus, if you have a set of match clauses that you want to reuse, a policy map simplifies your configuration, so you do not need to repeat the match clauses in each route map. You can use route maps that refer to policy lists in BGP.

Within a route map, you can include other match clauses in addition to policy lists. The policy list match clauses match on the incoming attributes only.

Policy lists support matching IPv4 addresses only; you cannot match IPv6 addresses.

For the match clauses in the policy map:

- Multiple match clauses are ANDed. That is, a route must satisfy each clause to match the policy list.
- Multiple values within a single match clause are ORed. That is, if a route matches any value within that match statement, it is considered to match the statement as a whole.

Before you begin

If you will configure match clauses for access lists, prefix lists, or AS path access lists, you must create those objects before you create the policy list.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon ( ) for the object.

To delete an unreferenced object, click the trash can icon ( ) for the object.

Step 4 Select **Policy List** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the policy list name in the first line of the CLI template (in the **policy-list** command).

Step 6 Click *action* in the **policy-list** command select one of the following:

- **permit**—Match. Connections that match this list are selected for the feature you are configuring.

- **deny**—Do not match. Connections that match this list are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this object to define which routes get redistributed, "denied" address spaces are simply not redistributed.

Step 7 Click **Show Disabled** above the template to expose the match commands. You must click the **+** icon to the left of the match statements you want to enable. Configure any combination of the following match statements to define the routes you are targeting.

- **match as-path**. Click the variable and select the AS Path objects that define the autonomous system numbers to match.
- **configure match ip address** *list-type*. Click the *list-type* variable and select whether you want to match the IP address in the route based on **access-list** or **prefix-list**. This will add a **match ip address** command, where you can click the variable and select either the standard access lists or IPv4 prefix lists that define the IP addresses to match.
- **configure match ip next-hop** *list-type*. Click the *list-type* variable and select whether you want to match the IP address of the next hop router in the route based on **access-list** or **prefix-list**. This will add a **match ip next-hop** command, where you can click the variable and select either the standard access lists or IPv4 prefix lists that define the IP addresses to match.
- **configure match ip route-source** *list-type*. Click the *list-type* variable and select whether you want to match the IP address of the route source in the route based on **access-list** or **prefix-list**. This will add a **match ip route-source** command, where you can click the variable and select either the standard access lists or IPv4 prefix lists that define the IP addresses to match.
- **match community** *community-list options*. Click the *community-list* variable and select the community list objects that define the communities to match. If you want routes to match the community list only if all communities in the list are matched, click *options* and select **exact-match**.
- **match interface**. Click the variable and select all of the interfaces in the routes to match.
- **match metric**. Click the variable and enter the routing multi-exit discriminator (MED) metric to match, from 1-4294967295.
- **match tag**. Click the variable and enter the route tag value to match, from 0-4294967295.

Step 8 Click **OK** to save the object.

You can now use the object in a route map object for use in BGP routing.

Configure Prefix Lists

Prefix lists are similar to access control lists. A prefix list is an ordered list of permit/deny rules, where permit indicates address prefixes that should match the list, and deny indicates address prefixes that should not match the list. The system evaluates matches top-down, and assigns the action based on the first matched rule, not necessarily the best matched rule. So, you need to specify sequence numbers carefully to ensure you get the matches you require.

You can use prefix lists for OSPF filtering or BGP, OSPF, or EIGRP route maps for route redistribution or injection, or for BGP neighbor filtering.

There are separate prefix lists for IPv4 and IPv6 addresses, but the structure of the lists are the same.

Procedure

Step 1 Click **View Configuration** in **Device** > **Advanced Configuration**.

Step 2 Select **Smart CLI** > **Objects** in the table of contents.

Step 3 Do one of the following:

- To create an object, click the + button.
- To edit an object, click the edit icon ( ) for the object.

To delete an unreferenced object, click the trash can icon ( ) for the object.

Step 4 Select **IPv4 Prefix List** or **IPv6 Prefix List** as the **CLI Template**.

Step 5 Enter a **Name** for the Smart CLI object. Note that this name is also entered as the Prefix List name in the first line of the CLI template (in the **prefix-list** command).

Step 6 Configure a prefix list entry, the **seq** command line.

Each entry is contained on a single line starting with the **seq** option.

a) In **seq**, click *sequence-number* and enter the number for this rule, from 1-4294967294. The number is relative to the sequence numbers for the other rules, with 1 being the first rule evaluated. Common practice is to skip count by 5, that is, 5, 10, 15, etc. This leaves you room to insert new rules without needing to change the sequence numbers of other rules.

b) Click *action* and select one of the following:

- **permit**—Match. Connections that match this rule are selected for the feature you are configuring.
- **deny**—Do not match. Connections that match this rule are excluded from the feature. Note that "denied" traffic is not dropped, it simply does not get the service applied to it. For example, in a route map, if you use this rule to define which routes get redistributed, "denied" address spaces are simply not redistributed.

c) Click *ip-address-mask* and enter the network address and mask (in CIDR format for IPv4) or prefix-length for IPv6. For example, 10.100.10.0/24 (IPv4) or 2001:DB8:0:CD30::/60 (IPv6).

The system uses an exact match for this address/mask unless you also include one of the **ge** or **le** options. For example, 10.100.10.10/8 does not match 10.100.10.0/24 unless you include **ge 9** in the rule.

The mask or prefix length can be:

- IPv4 = 0-32
- IPv6 = 0-128

d) (Optional.) You can use the **ge** and **le** keywords to specify the range of the prefix length to be matched for prefixes that are more specific than the IP address and mask/prefix length. Without these keywords, only exact matches are considered to match the rule.

ge *min-prefix-length* specifies the minimum prefix length to be matched. The minimum must be greater than the mask/prefix length and less than or equal to the maximum defined in the **le** option, if present.

le *max-prefix-length* specifies the maximum prefix length to be matched. The maximum must be greater than or equal to the minimum, if present, or greater than the mask/prefix length if the minimum value is not defined.

Besides the relative length limits mentioned above, the lengths in these options have the following outside limits:

- IPv4 = 1-32
- IPv6 = 0-128

Step 7 Add entries to complete the prefix list.

To add an entry, click **...** > **Duplicate** to the left of a **seq** line. A new entry is added immediately after the entry for which you click the Duplicate command.

For your convenience, it is best to try to keep the entries in sequence order. However, when deployed, the prefix list will be rewritten in the sequential order, even if you have them mixed up in the object.

Note that duplicating an entry simply inserts a new, empty entry, with no pre-configured characteristics. After creating the "duplicate," proceed as explained above to configure it for your needs.

Step 8 Click **OK** to save the object.

You can now use the object in a route map or routing process, or in a FlexConfig object, for a feature that requires a prefix list.

Examples

Following are some examples for how to match prefixes using a prefix list. The sequence number is left out of the examples for simplicity. The actual behavior of each rule is modified by any sequentially earlier rule that matches a subset of the covered address spaces.

- Deny the default route 0.0.0.0/0:

```
deny 0.0.0.0/0
```

- Permit the prefix10.0.0.0/8:

```
permit 10.0.0.0/8
```

- Accept a mask length of up to 24 bits in routes with the prefix 192/8:

```
permit 192.168.0.0/8 le 24
```

- Deny mask lengths greater than 25 bits in routes with a prefix of 192/8:

```
deny 192.168.0.0/8 ge 25
```

- Permit mask lengths from 8 to 24 bits in all address spaces:

```
permit 0.0.0.0/0 ge 8 le 24
```

- Deny mask lengths greater than 25 bits in all address spaces:

```
deny 0.0.0.0/0 ge 25
```

- Deny all routes with a prefix of 10/8:

```
deny 10.0.0.0/8 le 32
```

- Deny all masks with a length greater than 25 bits for routes with a prefix of 192.168.1/24:

```
deny 192.168.1.0/24 ge 25
```

- Permit all routes with a prefix of 0/0:

```
permit 0.0.0.0/0 le 32
```