

Activating Control Policy

A control policy enables the service provider to define certain actions that are performed during various subscriber life-cycle events. This chapter provides information about activating control policy on the BNG router. A control policy is defined using a policy-map. The policy-map contains a set of events - events during which certain actions are performed. The condition for performing an action is defined in a class-map. After a class-map is created, it is included in the policy-map. The policy-map is then activated on the router interface for the policy to take effect. One of the actions that can be performed by the policy map is activating dynamic template. The dynamic template is a container used to group a set of configuration items to be applied to a group of subscribers. This chapter covers the following topics:

- Control Policy Overview, on page 1
- Creating Class-Map, on page 3
- Creating Policy-Map, on page 4
- Activating Policy-Map, on page 8
- Defining Dynamic Templates, on page 9
- Additional References, on page 11

Control Policy Overview

A control policy enables the service provider to define actions that must be performed during various subscriber lifecycle events, such as creation of a session, connectivity loss, and so on. For the complete list of events, see Control Policy Events, on page 5.

Different actions can be executed for different subscribers based on various match criteria. Some actions that can be specified in the control policy are:

- Authenticating or authorizing a subscriber by an external AAA server
- Starting subscriber accounting
- Activating specific configurations on the subscriber using dynamic templates

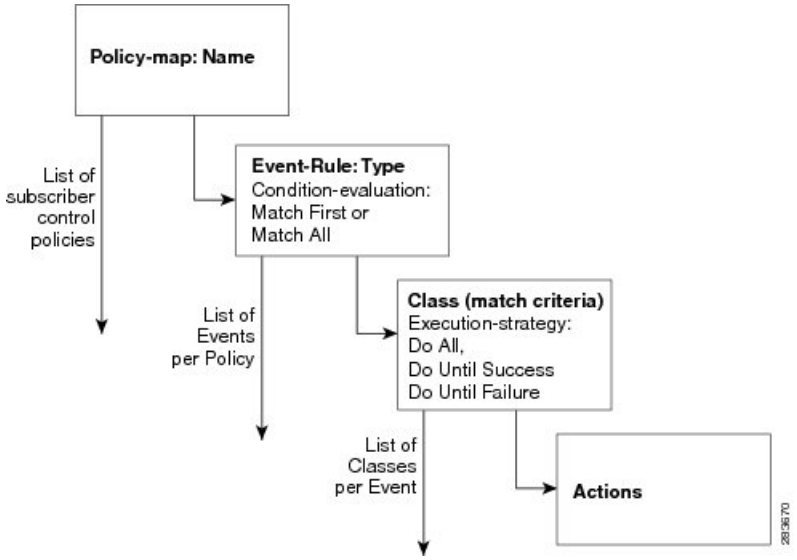
A control policy is deployed using policy-map and class-map. Each policy-map contains a list of events that the service provider considers applicable to the subscriber lifecycle. The policy-map also defines the actions that will be performed during these events. However, these actions are performed only when certain conditions are met. These conditions are called match criteria. The match criteria are defined in class-maps, which is included within the policy-map. It is possible to have different match criteria for different subscribers.

For example, a control policy can be created to start the "subscriber authentication" action, when a "session start" event occurs, for a specific "MAC address" match criteria. After this control policy is deployed, when the device having the specified MAC address starts a new session, BNG initiates the subscriber authentication process.

The actions defined in the policy-map are executed by action handlers. For more information about supported action handlers, see Action Handlers.

The following figure shows the structure of control policy. It illustrates that for each policy there can be multiple events; for each event, there can be multiple classes; and for each class, there can be multiple actions. As a result, a single policy map can be used to trigger multiple actions, when a match is found for a single or several criteria, during one or many events.

Figure 1: Control Policy



The following sample configuration shows the control policy structure:

```
policy-map type control subscriber policy-map-name
   event <event-type> [match-all|match-first]
      class type control subscriber <class-map-name>
           <seq#> <action-type> <action_options>
```

Note From Cisco IOS XR Software Release 5.2.2 and later, you can edit the class associated with the subscriber policy even while the sessions are active. Prior to this, new class map actions were not editable if the sessions were up, and any such dynamic policy-map changes resulted in clearing off the subscriber sessions.

Note Limit the `policy-map type control subscriber` number to 100.

Creating Class-Map

The class-map is used to define traffic class. The traffic is classified based on match criteria defined in the class-map. The parameter for match criteria can be protocol, MAC address, input interface, access group, and so on.

If more than one match criteria is listed in a single class-map, then instructions must be included defining how the match criteria are to be evaluated. The evaluation instruction are of two types:

- Match-any—A positive match is made if the traffic being evaluated matches any one of the specified criteria.
- Match-all—A positive match is made only if the traffic being evaluated matches all specified criteria.

Once a match is made, the traffic is considered as a member of the class.

Each class-map is assigned a name for identification. The class-map name is specified within the policy-map.

For creating a class-map, see Configuring a Class-Map, on page 3.

Configuring a Class-Map

Perform this task to configure a class-map for control policies. As an example, this class-map is created with the evaluation instruction, "match-any". The match criteria is "protocol" with value "PPP". As a result, a positive match is made when the session is uses PPP protocol.

SUMMARY STEPS

1. **configure**
2. **class-map type control subscriber match-any** *class-map-name*
3. **match protocol ppp**
4. **end-class-map**
5. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**class-map type control subscriber match-any** *class-map-name* **Example:**	Creates a new subscriber control class-map with a user defined name. Enters the class-map mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config)# class-map type control subscriber match-any clmap1`	Defines the match evaluation instruction to be "match-any".
Step 3	**match protocol ppp** **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# match protocol ppp`	Defines the match-criteria to be PPP protocol. **Note** More than one match statement can be applied per class-map.
Step 4	**end-class-map** **Example:** `RP/0/RSP0/CPU0:router(config-cmap)# end-class-map`	Ends the class map configuration.
Step 5	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring a Class-Map: An example

```
class-map type control subscriber match-any DHCP_class
match protocol dhcpv4
end-class-map
 !
!
end
```

Creating Policy-Map

The policy-map is used to define the events for which a set of actions are executed when a match, based on the class-map definitions, is made. For more information about the supported BNG events, see Control Policy Events, on page 5.

A policy-map lists a set of events. For each event, a set of class-maps are defined. For each class-map, a series of actions are listed sequentially. After the policy-map is applied on the BNG router interface, when the traffic matches the criteria mentioned in the class-map, the actions are performed.

If more than one class-map is listed in the policy-map, then instruction has to be specified that defines which class-maps should be applied. The evaluation instruction are of two types:

- First-match—Actions are performed only when a match is made for the first class-map.
- Match-all—Actions are performed for all matching classes.

Like with a class-map, each policy-map is assigned a name for identification. The policy-map name is specified when activating the policy-map on the router interface.

For creating a policy-map, see Configuring a Policy-Map, on page 6.

Control Policy Events

Control policy on BNG supports the events listed here. These events need to be defined while creating a policy-map using the task Configuring a Policy-Map, on page 6.

- Session-Start—This event is used by the PPPoE and DHCP access protocols to create a subscriber in the policy plane. The operator may configure the AAA actions and activate dynamic templates, suitable for subscriber.
- Session-Activate—Some access protocols require a two-stage session bring-up; for example, with PPPoE subscribers, the PPPoE Access protocol calls the Session-Start event for first sign of life (FSOL), followed by Session-Activate during PPP negotiation and authentication. The operator configures the AAA actions and activates the dynamic templates as suitable for the subscriber.
- Service-Stop—CoA is responsible for generating this event. The BNG operator configures the activate or deactivate actions, to put the subscriber in a default state when a service is stopped.
- Authentication-No-Response—If configured, this event is triggered when there is no response from the AAA server(s) for an authentication request. This event allows the network access server (NAS) operators to define how the failure should be handled. If the authentication-no-response event is not configured, then the authentication failure result is propagated to the access protocol for default handling.
- Authorization-No-Response—If configured, this event is triggered when there is no response from the AAA server(s) for an authorization request. This event allows the NAS operators to define how the failure should be handled. If the authorization-no-response event is not configured, then the authorization results are propagated to the access protocol for default handling, which causes the client who triggered the authorization to disconnect the subscriber session.
- Authentication-Failure—If configured and if the RADIUS server returns an authentication failure, then the Policy Rule Engine returns an "Authentication-Success" to the client that originated the request, in order to prevent it from disconnecting the subscriber. Furthermore, instead of depending on the client to provide the necessary behavior, the actions within the configured Authentication-Failure event are applied on the subscriber.
- Authorization-Failure—The authorization failure event indicates a RADIUS server rejection for the access request. If configured, the service provider overrides the default handling of the failure from the client.
- Timed-Policy-Expiry—If configured, this event is triggered as a result of a policy set-timer action that is configured and set on a subscriber session. This event allows NAS operators to define a timer for a number of possible scenarios. The set timer indicates that certain subscriber state changes have taken place. If sessions are not in the desired state, the NAS operators can disconnect or terminate the session through a configured disconnect action, or impose a different user policy.
- Account-Logon—If configured, this event provides an override behavior to the default account-logon processing. The default behavior only triggers authentication with provided credentials. However, if you

override the default account-logon event, then you must explicitly configure the authentication action, and any additional action you require.

- Account-Logoff—If configured, this event provides an override behavior for the default account-logoff processing. The default behavior of the account-logoff processing is to disconnect the subscriber. Being able to override the default behavior is useful. Instead of disconnecting the subscriber, the service provider can perform a re-authentication. The re-authentication is done through a new account-logon by enabling HTTP Redirect feature on the subscriber.
- Idle-Timeout—If configured, this event terminates the IPoE and PPPoE subscriber sessions when the timeout period expires. The default behavior of the Idle-Timeout event is to disconnect the session. You can configure a **monitor** action under the idle timeout event for a subscriber policy, to prevent the termination of the subscriber session when the idle timeout period expires.

Configuring a Policy-Map

Perform this task to configure policy map for control policies. As an example, this policy-map is created for the Session-Start and Session-Activate events. For the Session-Start event, a dynamic template is activated. For the Session-Activate event, an authentication process is invoked. For more information about the supported events, see Control Policy Events, on page 5.

SUMMARY STEPS

1. **configure**
2. **policy-map type control subscriber** *policy-map-name*
3. **event session-start match-all**
4. **class type control subscriber** *class_name* **do-until-failure**
5. *sequence_number* **activate dynamic-template** *dynamic-template_name*
6. **event session-activate match-all**
7. **class type control subscriber** *class_name* **do-until-failure**
8. *sequence_number* **authenticate aaa list default**
9. **end-policy-map**
10. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map type control subscriber** *policy-map-name* **Example:**	Creates a new policy-map with user-defined name. Enters the policy-map mode.

	Command or Action	Purpose
	`RP/0/RSP0/CPU0:router(config)# policy-map type control subscriber plmap1`	
Step 3	**event session-start match-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# event session-start match-all`	Defines an event (session start) for which actions will be performed. Defines the match instruction to be "match-all", which executes actions for all matched classes.
Step 4	**class type control subscriber** *class_name* **do-until-failure** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-e)# class type control subscriber CL1 do-until-failure`	Associates a class-map with the event. The class-map name has to be specified. Instructs that the actions will be performed until a failure occurs.
Step 5	*sequence_number* **activate dynamic-template** *dynamic-template_name* **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 1 activate dynamic-template template1`	Defines the action to be performed. In this case, it activates a dynamic-template. **Note** This command can be repeated to define multiple actions.
Step 6	**event session-activate match-all** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# event session-activate match-all`	Defines an event (activate session) for which actions will be performed. Defines the match instruction to be "match-all", which executes actions for all matched classes.
Step 7	**class type control subscriber** *class_name* **do-until-failure** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-e)# class type control subscriber CL1 do-until-failure`	Associates a class-map with the event. The class-map name needs to be specified. Instructs that the actions will be performed until a failure occurs.
Step 8	*sequence_number* **authenticate aaa list default** **Example:** `RP/0/RSP0/CPU0:router(config-pmap-c)# 2 authenticate aaa list default`	Defines the action to be performed. In this case, it initiates the authentication of AAA list.
Step 9	**end-policy-map** **Example:** `RP/0/RSP0/CPU0:router(config-pmap)# end-policy-map`	Ends the policy map configuration.

	Command or Action	Purpose
Step 10	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Configuring a Policy-Map: An example

```
policy-map type control subscriber PL1
 event session-start match-first
  class type control subscriber DHCP_class do-until-failure
   1 activate dynamic-template dhcp
  class type control subscriber class-default do-until-failure
! Packet trigger is default
   1 activate dynamic-template packet-trigger
 end-policy-map
 !
!
end


\\Configuring a Policy-Map with idle-timeout event and monitor action

policy-map type control subscriber PL2
  event idle-timeout
   class type control subscriber DHCP_class
    1 monitor
```

Activating Policy-Map

After a policy-map is created, it needs to be activated on a router interface. The policies are implemented only after the policy-map activation is completed. One or more policy-maps will constitute the service-policy. To enable the service-policy, see Enabling a Service-Policy on a Subscriber Interface, on page 8.

Enabling a Service-Policy on a Subscriber Interface

Perform this task to enable a service-policy on a subscriber interface. The process involves attaching a previously created policy-map with an interface. Once this process is complete, the actions defined in the class-map will take effect for the traffic coming on the interface on which service-policy is enabled.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **service-policy type control subscriber** *policy_name*
4. Use the **commit** or **end** command.

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface Bundle-Ether100.10`	Enters the interface configuration mode for the bundle-ether access interface. **Note** For IPoE sessions, it is recommended that Dynamic ARP learning be disabled in the access-interface, using the **arp learning disable** command.
Step 3	**service-policy type control subscriber** *policy_name* **Example:** `RP/0/RSP0/CPU0:router(config-if)# service-policy type control subscriber plmap1`	Applies a pre-defined policy-map named 'plmap1' to an access interface.
Step 4	Use the **commit** or **end** command.	**commit** —Saves the configuration changes and remains within the configuration session. **end** —Prompts user to take one of these actions: • **Yes** — Saves configuration changes and exits the configuration session. • **No** —Exits the configuration session without committing the configuration changes. • **Cancel** —Remains in the configuration session, without committing the configuration changes.

Defining Dynamic Templates

A dynamic template is a container used to group a set of configuration settings, and apply them to the subscriber sessions. A dynamic template is globally configured through CLI. However, defining the dynamic template does not immediately cause the configuration to be applied to a subscriber interface. The configuration within

a dynamic template is applied to a subscriber interface, only when the dynamic template is activated using a control policy. Similarly, the applied configurations are stopped, only when the dynamic template is deactivated using a control policy.

There are three basic types of dynamic-templates:

- PPP templates—It contains specific configurations related to the PPPoE protocol.
- IP Subscriber templates—It contains specific configurations that are activated on IP subscriber sessions.
- Service templates—It contains service-related configuration that are activated in response to session life-cycle events. Service templates are precluded from containing interface or media-specific commands.

A dynamic template can either be configured on the CLI, or downloaded from the AAA server. In the following sample configuration, the policy map activates an IP Subscriber dynamic template that is defined on the CLI.

```
dynamic-template
type ipsubscriber ipsub
ipv4 unnumbered Loopback400
policy-map type control subscriber PL2
event session-start match-first
class type control subscriber class-default do-all
1 activate dynamic-template ipsub
```

There are two types of dynamic templates that are downloaded from the AAA server—user profiles and service profiles. User profiles are applied to a single subscriber, whereas, service profiles can be applied to multiple subscribers. In the following sample configuration, the policy map downloads a service template from the AAA server.

```
Radius Config:
service1 Password="xxxxxx"
Cisco-avpair = "ipv4:ipv4-unnumbered=Loopback400"

Router Config:
policy-map type control subscriber PL2
event session-start match-first
class type control subscriber class-default do-all
1 activate dynamic-template service1 aaa list default
```

In the above example, the "aaa list default" keyword specifies that the template "service1" be downloaded from the AAA server. A template is downloaded only once. If there are multiple control policies referring to service1, then those will get the previously downloaded version.

It is possible to activate more than one dynamic template on the same subscriber interface, for the same event or different events. If the configurations for a particular functionality is defined in multiple dynamic templates, the configurations are derived from all the templates on a certain order of precedence. This order is based on the type of dynamic template, and whether it is being applied from CLI or AAA. The order is:

- Template applied by the user profile from AAA
- Template applied by the service profile from AAA
- IP Subscriber template applied from CLI
- PPP template applied from CLI
- Service template applied from CLI

The tasks involving the use of dynamic templates to define specific feature configurations are included in their corresponding feature topics.

Additional References

These sections provide references related to implementing control policy.

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html