Configuring Global Policies

About Global Policies

The APIC fabric has many fabric level configurations, which are applied to the entire fabric components (switches and ports). In some cases, lower level policies (switch or interface level) exist to override these policies. For example, while MCP policy can enable the MCP feature globally, an interface level MCP policy exists to enable or disable MCP on an individual interface.

Configuring Out-of-Band Management NTP

When an ACI fabric is deployed with out-of-band management, each node of the fabric is managed from outside the ACI fabric. You can configure an out-of-band management NTP server so that each node can individually query the same NTP server as a consistent clock source.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters configuration mode.
Step 2	**template ntp-fabric** *ntp-fabric-template-name* **Example:** `apic1(config)# template ntp-fabric pol1`	Specifies the NTP template (policy) for the fabric.
Step 3	[**no**] **server** *dns-name-or-ipaddress* [**prefer**] [**use-vrf** {**inband-mgmt** | **oob-default**}] [**key** *key-value*] **Example:** `apic1(config-template-ntp-fabric)# server 192.0.20.123 prefer use-vrf oob-mgmt`	Configures an NTP server for the active NTP policy. To make this server the preferred server for the active NTP policy, include the **prefer** keyword. If NTP authentication is enabled, specify a reference key ID. To specify the in-band or out-of-band management access VRF, include the **use-vrf** keyword with the **inb-default** or **oob-default** keyword.
Step 4	[**no**] **authenticate** **Example:** `apic1(config-template-ntp-fabric)# no authenticate`	Enables (or disables) NTP authentication.
Step 5	[**no**] **authentication-key** *key-value* **Example:** `apic1(config-template-ntp-fabric)# authentication-key 12345`	Configures an authentication NTP authentication. The range is 1 to 65535.
Step 6	[**no**] **trusted-key** *key-value* **Example:** `apic1(config-template-ntp-fabric)# trusted-key 54321`	Configures a trusted NTP authentication. The range is 1 to 65535.
Step 7	**exit** **Example:** `apic1(config-template-ntp-fabric)# exit`	Returns to global configuration mode
Step 8	**template pod-group** *pod-group-template-name* **Example:** `apic1(config)# template pod-group allPods`	Configures a pod-group template (policy).

	Command or Action	Purpose
Step 9	**inherit ntp-fabric** *ntp-fabric-template-name* **Example:** `apic1(config-pod-group)# inherit ntp-fabric pol1`	Configures the NTP fabric pod-group to use the previously configured NTP fabric template (policy).
Step 10	**exit** **Example:** `apic1(config-template-pod-group)# exit`	Returns to global configuration mode
Step 11	**pod-profile** *pod-profile-name* **Example:** `apic1(config)# pod-profile all`	Configures a pod profile.
Step 12	**pods** {*pod-range-1-255* | **all**} **Example:** `apic1(config-pod-profile)# pods all`	Configures a set of pods.
Step 13	**inherit pod-group** *pod-group-name* **Example:** `apic1(config-pod-profile-pods)# inherit pod-group allPods`	Associates the pod-profile with the previously configured pod group.
Step 14	**end** **Example:** `apic1(config-pod-profile-pods)# end`	Returns to EXEC mode.

Examples

This example shows how to configure a preferred out-of-band NTP server and how to verify the configuration and deployment.

```
apic1# configure t
apic1(config)# template ntp-fabric pol1
apic1(config-template-ntp-fabric)# server 192.0.20.123 use-vrf oob-default
apic1(config-template-ntp-fabric)# no authenticate
apic1(config-template-ntp-fabric)# authentication-key 12345
apic1(config-template-ntp-fabric)# trusted-key 12345
apic1(config-template-ntp-fabric)# exit
apic1(config)# template pod-group allPods
apic1(config-pod-group)# inherit ntp-fabric pol1
apic1(config-pod-group)# exit
apic1(config)# pod-profile all
apic1(config-pod-profile)# pods all
apic1(config-pod-profile-pods)# inherit pod-group allPods
apic1(config-pod-profile-pods)# end
apic1#


apic1# show ntpq
```

```
nodeid     remote        refid   st    t   when   poll   reach  delay    offset   jitter

------  -  ------------  ------  ----  --  -----  -----  -----  ------   ------   ------
1       *  192.0.20.123  .GPS.         u   27     64     377    76.427   0.087    0.067

2       *  192.0.20.123  .GPS.         u   3      64     377    75.932   0.001    0.021
3       *  192.0.20.123  .GPS.         u   3      64     377    75.932   0.001    0.021
```

Configuring the System Clock

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **clock display-format** {**local** | **utc**} **Example:** `apic1(config)# clock display-format local`	Sets the clock date time format to either local or UTC time.
Step 3	[**no**] **clock show-offset enable** **Example:** `apic1(config)# clock show-offset enable`	Enables (or disables) the display of the offfset from UTC. This setting is valid only when the display-format is local.
Step 4	[**no**] **clock timezone** *timezone-code* **Example:** `apic1(config)# clock timezone n420_America-Los_Angeles`	Specifies the local time zone. The default is p0_utc.
Step 5	**show clock** **Example:** `apic1(config)# show clock`	Specifies the delay time for LLDP to initialize on any interface . The range is 1 to 10 seconds; the default is 2 seconds.

Examples

This example shows how to configure the system clock for local time in the Los Angeles timezone.

```
apic1# configure terminal
apic1(config)# clock display-format local
apic1(config)# clock show-offset enable
apic1(config)# clock timezone n420_America-Los_Angeles
apic1(config)# show clock
Time : 20:47:37.038 UTC-08:00 Sun Nov 08 2015
```

Configuring Error Disable Recovery

The error disabled recovery (EDR) policy is a fabric level policy that can enable ports that loop detection and BPDU policies disabled after an interval that the administrator can configure.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **errdisable recovery interval** *seconds* **Example:** `apic1(config)# errdisable recovery interval 300`	Specifies the interval for an interface to recover from the error-disabled state. The range is from 30 to 65535 seconds
Step 3	[**no**] **errdisable recovery cause** {**bpduguard** | **ep-move** | **mcp-loop**} **Example:** `apic1(config)# errdisable recovery cause mcp-loop`	Specifies a condition under which the interface automatically recovers from the error-disabled state, and the device retries bringing the interface up. The default is disabled. The condition options are: • **bpduguard** —Enable timer to recover from a BPDU guard error disable. • **ep-move** —Enable timer to recover from an endpoint move error disable. • **mcp-loop** —Enable timer to recover from an MCP loop error disable. • **storm-control-recovery** —Enable timer to recover from a storm control recovery error disable.

Examples

This example shows how to configure EDR to recover from an MCP loop error disable.

```
apic1# configure terminal
apic1(config)# errdisable recovery interval 300
apic1(config)# errdisable recovery cause mcp-loop
```

Configuring Link Level Discovery Protocol

The Link Layer Discovery Protocol (LLDP) is a device discovery protocol that allows network devices to advertise information about themselves to other devices on the network. LLDP determines the layer 2 connectivity between switches.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **lldp holdtime** *seconds* **Example:** `apic1(config)# lldp holdtime`	Specifies the hold time to be sent in LLDP packets.
Step 3	[**no**] **lldp holdtime** *seconds* **Example:** `apic1(config)# lldp holdtime 120`	Specifies the hold time to be sent in LLDP packets. The range is 10 to 255 seconds; the default is 120 seconds.
Step 4	[**no**] **lldp reinit** *seconds* **Example:** `apic1(config)# lldp reinit 2`	Specifies the delay time for LLDP to initialize on any interface . The range is 1 to 10 seconds; the default is 2 seconds.
Step 5	[**no**] **lldp timer** *seconds* **Example:** `apic1(config)# lldp timer 30`	Specifies the transmission frequency seconds of LLDP updates in seconds. The range is 5 to 254 seconds; the default is 30 seconds.

Examples

This example shows how to configure LLDP.

```
apic1# configure terminal
apic1(config)# lldp holdtime 120
apic1(config)# lldp reinit 2
apic1(config)# lldp timer 30
```

Configuring Miscabling Protocol

The ACI fabric provides loop detection policies that can detect loops in Layer 2 network segments that are connected to ACI access ports. The ACI fabric implements the mis-cabling protocol (MCP), a fabric level policy that allows provisioning of MCP parameters as well as determining the port behavior if mis-cabling is

detected. MCP works in a complementary manner with STP that is running on external Layer 2 networks, and handles Bridge Protocol Data Unit (BPDU) packets that access ports receive.

A fabric administrator provides a key that MCP uses to identify which MCP packets are initiated by the ACI fabric. The administrator can choose how the MCP policies identify loops and how to act upon the loops: syslog only, or disable the port.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **mcp action port-disable** **Example:** `apic1(config)# mcp action port-disable`	Specifies whether a port should be place in a disabled state if mis-cabling is detected.
Step 3	[**no**] **mcp enable** [**key** *key-value*] **Example:** `apic1(config)# mcp enable key 0123456789abcdef`	Allows enabling or disabling of the MCP protocol globally for the entire fabric. A password (key) is required when enabling the policy but not when disabling.
Step 4	[**no**] **mcp factor** *number* **Example:** `apic1(config)# mcp factor 64`	Sets the loop detection multiplication factor, which is used while sending MCP packets. The range is 1 to 255.
Step 5	[**no**] **mcp init-delay** *seconds* **Example:** `apic1(config)# mcp init-delay 180`	Specifies the initial delay time. The range is 0 to 1800 seconds; the default is 180.
Step 6	[**no**] **mcp transmit-frequency** *frequency* **Example:** `apic1(config)# mcp transmit-frequency 2`	Sets the frequency of transmission of MCP packets to detect mis-cabling. The range is 100 milliseconds to 300 seconds; the default is 2 seconds.

Examples

This example shows how to configure MCP for a transmit frequency of 2 seconds.

```
apic1# configure terminal
apic1(config)# mcp action port-disable
apic1(config)# mcp enable key 0123456789abcdef
apic1(config)# mcp factor 64
apic1(config)# mcp init-delay 180
apic1(config)# mcp transmit-frequency 2
```

This example shows how to configure MCP for a transmit frequency of 2 seconds and 300 milliseconds.

```
apic1# configure terminal
apic1(config)# mcp action port-disable
apic1(config)# mcp enable key 0123456789abcdef
apic1(config)# mcp factor 64
apic1(config)# mcp init-delay 180
apic1(config)# mcp transmit-frequency 2 300
```

Configuring the Endpoint Loop Protection Policy

The endpoint loop protection policy is a fabric level policy used in detection of frequent endpoint (host) moves from one fabric port to another. The policy configures what action is to be taken if such an event is detected.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **endpoint loop-detect action** {**bd-learn-disable** | **port-disable**} **Example:** `apic1(config)# endpoint loop-detect action port-disable`	Specifies the action to perform when an endpoint loop is detected. The options are: • **bd-learn-disable** —Disable MAC address learning on the bridge domain. • **port-disable** —Disable the port.
Step 3	[**no**] **endpoint loop-detect enable** **Example:** `apic1(config)# endpoint loop-detect enable`	Allows enabling or disabling of the endpoint loop protection protocol globally for the entire fabric.
Step 4	[**no**] **endpoint loop-detect factor** *number* **Example:** `apic1(config)# endpoint loop-detect factor 64`	Sets the loop detection multiplication factor. The range is 1 to 255.
Step 5	[**no**] **endpoint loop-detect interval** *seconds* **Example:** `apic1(config)# endpoint loop-detect interval 60`	Specifies the loop detection interval. The range is 30 to 300 seconds.

Examples

This example shows how to configure the endpoint loop protection policy.

```
apic1# configure terminal
apic1(config)# endpoint loop-detect action port-disable
apic1(config)# endpoint loop-detect enable
apic1(config)# endpoint loop-detect factor 64
apic1(config)# endpoint loop-detect interval 60
```

Configuring the Rogue Endpoint Control Policy

About the Rogue Endpoint Control Policy

A rogue endpoint attacks top of rack (ToR) switches through frequently, repeatedly injecting packets on different ToR ports and changing 802.1Q tags (thus, emulating endpoint moves) causing learned class and EPG port changes. Misconfigurations can also cause frequent IP and MAC address changes (moves).

Such rapid movement in the fabric causes significant network instability, high CPU usage, and in rare instances, endpoint mapper (EPM) and EPM client (EPMC) crashes due to significant and prolonged messaging and transaction service (MTS) buffer consumption. Also, such frequent moves may result in the EPM and EPMC logs rolling over very quickly, hampering debugging for unrelated endpoints.

The rogue endpoint control feature addresses this vulnerability by quickly:

- Identifying such rapidly moving MAC and IP endpoints.
- Stopping the movement by temporarily making endpoints static (thus, quarantining the endpoint).
- Prior to 3.2(6) release: Keeping the endpoint static for the **Rogue EP Detection Interval** and dropping the traffic to and from the rogue endpoint. After this time expires, deleting the unauthorized MAC or IP address.
- In the 3.2(6) release and later: Keeping the endpoint static for the **Rogue EP Detection Interval** (this feature no longer drops the traffic). After this time expires, deleting the unauthorized MAC or IP address.
- Generating a host tracking packet to enable the system to re-learn the impacted MAC or IP address.
- Raising a fault, to enable corrective action.

The rogue endpoint control policy is configured globally and, unlike other loop prevention methods, functions at the level of individual endpoints (IP and MAC addresses). It does not distinguish between local or remote moves; any type of interface change is considered a move in determining if an endpoint should be quarantined.

The rogue endpoint control feature is disabled by default.

Configure Rogue Endpoint Control Using the NX-OS Style CLI

You can configure the **Rogue EP Control** policy for the fabric, to detect and delete unauthorized endpoints, using the NX-OS style CLI.

Procedure

Step 1 **configure**

Enters global configuration mode.

Example:

```
apic1# configure
```

Step 2 **endpoint rogue-detect enable**

Enables the global Rogue Endpoint Control policy.

Example:

```
apic1(config)# endpoint rogue-detect enable
```

Step 3 **endpoint rogue-detect hold-interval** *hold_interval*

Sets the hold interval in seconds after the endpoint is declared rogue, where it is kept static so learning is prevented, and the traffic to and from the rogue endpoint is dropped. After this interval, the endpoint is deleted. Valid values are from 1800 to 3600 seconds. The default is 1800.

Example:

```
apic1(config)# endpoint rogue-detect hold-interval 1800
```

Step 4 **endpoint rogue-detect interval** *interval*

Sets the rogue detection interval in seconds, which specifies the time to detect rogue endpoints. Valid values are from 0 to 65535 seconds. The default is 60.

Example:

```
apic1(config)# endpoint rogue-detect interval 60
```

Step 5 **endpoint rogue-detect factor** *factor*

Specifies the multiplication factor for determining if an endpoint is unauthorized. If the endpoint moves more times during the interval, the EP is declared rogue. Valid values are from 2 to 10. The default is 6.

Example:

```
apic1# endpoint rogue-detect factor 6
```

Step 6 This example configures a Rogue Endpoint Control policy.

Example:

```
apic1# cconfigure
apic1(config)# endpoint rogue-detect enable
apic1(config)# endpoint rogue-detect hold-interval 1800
apic1(config)# endpoint rogue-detect interval 60
apic1(config)# endpoint rogue-detect factor 6
```

Configuring IP Aging

Overview

The IP Aging policy tracks and ages unused IP addresses on an endpoint. Tracking is performed using the endpoint retention policy configured for the bridge domain to send ARP requests (for IPv4) and neighbor solicitations (for IPv6) at 75% of the local endpoint aging interval. When no response is received from an IP address, that IP address is aged out.

This document explains how to configure the IP Aging policy.

Configuring the IP Aging Policy Using the NX-OS-Style CLI

This section explains how to enable and disable the IP Aging policy using the CLI.

Procedure

Step 1 To enable the IP aging policy:

Example:

```
ifc1(config)# endpoint ip aging
```

Step 2 To disable the IP aging policy:

Example:

```
ifav9-ifc1(config)# no endpoint ip aging
```

What to do next

To specify the interval used for tracking IP addresses on endpoints, create an Endpoint Retention policy.

Configuring the Dynamic Load Balancer

Dynamic load balancing (DLB) adjusts the traffic allocations according to congestion levels. DLB measures the congestion across the available paths and places the flows on the least congested paths, which results in an optimal or near optimal placement of the data.

DLB can be configured to place traffic on the available uplinks using the granularity of flows or flowlets. Flowlets are bursts of packets from a flow that are separated by suitably large gaps in time. If the idle interval between two bursts of packets is larger than the maximum difference in latency among available paths, the second burst (or flowlet) can be sent along a different path than the first without reordering packets. This idle interval is measured with a timer called the flowlet timer. Flowlets provide a higher granular alternative to flows for load balancing without causing packet reordering.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **system dynamic-load-balance mode** {**dynamic-aggressive** | **dynamic-conservative** | **link-failure-resiliency** | **packet-prioritization**}	Specifies the mode of operation of the load balancer. The modes are: • **dynamic-aggressive** —The flowlet timeout is a relatively small value. This very fine-grained dynamic load balancing is optimal for the distribution of traffic, but some packet reordering might occur. However, the overall benefit to application performance is equal to or better than the conservative mode. • **dynamic-conservative** —The flowlet timeout is a larger value that guarantees packets are not to be re-ordered. The tradeoff is less granular load balancing because new flowlet opportunities are less frequent. • **link-failure-resiliency** —Static load balancing gives a distribution of flows across the available links that is roughly even. • **packet-prioritization** —Dynamic Packet Prioritization (DPP) prioritizes short flows higher than long flows; a short flow is less than approximately 15 packets. Because short flows are more sensitive to latency than long ones, DPP can improve overall application performance. `apic1(config)# system dynamic-load-balance mode packet-prioritization`

Examples

This example shows how to configure dynamic load balancing with packet prioritization.

```
apic1# configure terminal
apic1(config)# system dynamic-load-balance mode packet-prioritization
```

Configuring Spanning Tree Protocol

Multiple spanning-tree (MST) enables multiple VLANs to be mapped to the same spanning-tree instance, reducing the number of spanning-tree instances needed to support a large number of VLANs.

Note Multiple Spanning Tree (MST) is not supported on interfaces configured with the Per Port VLAN feature (configuring multiple EPGs on a leaf switch using the same VLAN ID with localPort scope).

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	**spanning-tree mst configuration** **Example:** `apic1(config)# spanning-tree mst configuration`	Enters global configuration mode.
Step 3	[**no**] **bpdu-filter** **Example:** `apic1(config-stp)# bpdu-filter`	Enters global configuration mode.
Step 4	[**no**] **region** *region-name* **Example:** `apic1(config-stp)# region region1`	For switches to participate in multiple spanning-tree (MST) instances, you must consistently configure the switches with the same MST configuration information. A collection of interconnected switches that have the same MST configuration comprises an MST region. Each region can support up to 65 spanning-tree instances.
Step 5	[**no**] **instance** *instance-id* **vlan** *vlan-range* **Example:** `apic1(config-stp-region)# instance 2 vlan 1-63`	Maps VLANs to an MST instance. You can assign a VLAN to only one spanning-tree instance at a time. The instance ID range is 1 to 4094. To specify a VLAN range, use a hyphen.
Step 6	**revision** *number* **Example:** `apic1(config-stp-region)# revision 16`	Specifies the configuration revision number. The range is 0 to 65535.

Examples

This example shows how to configure an MST spanning-tree policy.

```
apic1# configure terminal
apic1(config)# spanning-tree mst configuration
apic1(config-stp)# bpdu-filter
apic1(config-stp)# region region1
apic1(config-stp-region)# instance 2 vlan 1-63
apic1(config-stp-region)# revision 16
```

Configuring IS-IS

Intermediate System-to-Intermediate System (IS-IS) is a dynamic link-state routing protocol that can detect changes in the network topology and calculate loop-free routes to other nodes in the network.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	**template isis-fabric** *isis-fabric-template-name* **Example:** `apic1(config)# template isis-fabric polIsIs`	Enters Intermediate System-to-Intermediate System (IS-IS) configuration mode and creates an IS-IS fabric template (policy).
Step 3	[**no**] **lsp-fast-flood** **Example:** `apic1(config-template-isis-fabric)# lsp-fast-flood`	Enables the fast-flood feature, which improves convergence time when new link-state packets (LSPs) are generated in the network and shortest path first (SPF) is triggered by the new LSPs. We recommend that you enable the fast-flooding of LSPs before the router runs the SPF computation, to ensure that the whole network achieves a faster convergence time.
Step 4	[**no**] **lsp-gen-interval level-1** *lsp-max-wait* [*lsp-initial-wait lsp-second-wait*] **Example:** `apic1(config-template-isis-fabric)# lsp-gen-interval level-1 500 500 500`	Configures the IS-IS throttle for LSP generation. The parameters are as follows: • *lsp-max-wait* —The maximum wait between the trigger and LSP generation. • *lsp-initial-wait* —The initial wait between the trigger and LSP generation.

	Command or Action	Purpose
		• *lsp-second-wait* —The second wait used for LSP throttle during backoff. The *lsp-max-wait* parameter is required. The other two parameters are optional but must appear together. The range for each is 50 to 120000 milliseconds.
Step 5	[**no**] **lsp-mtu** *mtu* **Example:** `apic1(config-template-isis-fabric)# lsp-mtu 2048`	Sets the maximum transmission unit (MTU) size of IS-IS hello packets. The range is 256 to 4352. IS-IS hello packets are used to discover and maintain adjacencies. By default, the hello packets are padded to the full maximum transmission unit (MTU) size to allow for early detection of errors due to transmission problems with large frames or due to mismatched MTUs on adjacent interfaces. However, IS-IS adjacency formation may fail due to MTU mismatch on a link, requiring the adjustment of the MTU size.
Step 6	[**no**] **spf-interval level-1** *spf-max-wait* [*spf-initial-wait spf-second-wait*] **Example:** `apic1(config-template-isis-fabric)# spf-interval level-1 500 500 500`	Configures the interval between LSA arrivals. The parameters are as follows: • *spf-max-wait* —The maximum wait between the trigger and SPF computation. • *spf-initial-wait* —The initial wait between the trigger and SPF computation. • *spf-second-wait* —The second wait used for SPF computation during backoff. The *spf-max-wait* parameter is required. The other two parameters are optional but must appear together. The range for each is 50 to 120000 milliseconds.
Step 7	**exit** **Example:** `apic1(config-template-isis-fabric)# exit`	Returns to global configuration mode.
Step 8	**template pod-group** *pod-group-template-name* **Example:** `apic1(config)# template pod-group allPods`	Creates a pod group template (policy).

	Command or Action	Purpose
Step 9	**inherit pod-group** *pod-group-name* **Example:** `apic1(config-pod-group)# inherit isis-fabric polIsIs`	Configures the template pod-group to use the previously configured isis-fabric template (policy).
Step 10	**exit** **Example:** `apic1(config-pod-group)# exit`	Returns to global configuration mode.
Step 11	**pod-profile** *pod-profile-name* **Example:** `apic1(config)# pod-profile all`	Configures a pod profile.
Step 12	**pods** {*pod-range-1-255* | **all**} **Example:** `apic1(config-pod-profile)# pods all`	Configures a set of pods.
Step 13	**inherit pod-group** *pod-group-name* **Example:** `apic1(config-pod-profile-pods)# inherit pod-group allPods`	Configures the pod-profile to use the previously configured pod group.
Step 14	**end** **Example:** `apic1(config-pod-profile-pods)# end`	Returns to EXEC mode.

Examples

This example shows how to configure IS-IS.

```
aapic1# configure
apic1(config)# template isis-fabric polIsIs
apic1(config-template-isis-fabric)# lsp-fast-flood
apic1(config-template-isis-fabric)# lsp-gen-interval level-1 500 500 500
apic1(config-template-isis-fabric)# lsp-mtu 2048
apic1(config-template-isis-fabric)# spf-interval level-1 500 500 500
apic1(config-template-isis-fabric)# exit
apic1(config)# template pod-group allPods
apic1(config-pod-group)# inherit isis-fabric polIsIs
apic1(config-pod-group)# exit
apic1(config)# pod-profile all
apic1(config-pod-profile)# pods all
apic1(config-pod-profile-pods)# inherit pod-group allPods
apic1(config-pod-profile-pods)# end
apic1#
```

Configuring BGP Route Reflectors

The ACI fabric route reflectors use multiprotocol Border Gateway Protocol (MP-BGP) to distribute external routes within the fabric. To enable route reflectors in the ACI fabric, the fabric administrator must select the spine switches that will be the route reflectors, and provide the autonomous system (AS) number. For redundancy purposes, more than one spine is configured as a router reflector node (one primary and one secondary reflector).

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	**bgp-fabric** **Example:** `apic1(config)# bgp-fabric`	Enters BGP configuration mode for the fabric.
Step 3	**asn** *asn-value* **Example:** `apic1(config-bgp-fabric)# asn 123456789`	Configures the BGP Autonomous System number (ASN), which uniquely identifies an autonomous system. The ASN is between 1 and 4294967295. We recommend that you enable the fast-flooding of LSPs before the router runs the SPF computation, to ensure that the whole network achieves a faster convergence time.
Step 4	[**no**] **route-reflector spine** *list* **Example:** `apic1(config-bgp-fabric)# route-reflector spine spine1,spine2`	Configure up to two spine nodes as route reflectors. For redundancy ,you should configure primary and secondary route reflectors.

Examples

This example shows how to configure spine1 and spine2 as BGP route reflectors.

```
apic1# configure
apic1(config)# bgp-fabric
apic1(config-bgp-fabric)# asn 123456789
apic1(config-bgp-fabric)# route-reflector spine spine1,spine2
apic1(config-bgp-fabric)# exit
apic1(config)#
```

Decommissioning a Node

Two levels of decommissioning are supported:

- Regular—Similar to disabling the node. After being decommissioned, the node cannot rejoin the fabric until the **no decommission** command is executed.
- Complete—When the node is decommissioned, all fabric configuration related to the node is cleared.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **decommission** {**controller** | **switch**} *node-id* [**remove-from-controller**] **Example:** `apic1(config)# decommission switch 104 remove-from-controller`	Decommissions the specified node. Note that controller node ID numbers are between 1 and 100, while switch node ID numbers are between 101 and 4000.

Examples

This example shows how to perform a complete decommissioning of node 104 (a switch) and recommission node 5 (a controller), which was decommissioned with the regular level.

```
apic1# configure
apic1(config)# decommission switch 104 remove-from-controller
apic1(config)# no decommission controller 5
```

Configuring Power Management

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **power redundancy-policy** *policy-name* **Example:**	Creates or configures a power supply redundancy policy.

	Command or Action	Purpose
	`apic1(config)# power redundancy-policy myPowerPolicy`	
Step 3	[**no**] **description** *text* **Example:** `apic1(config-power)# description 'This is my power redundancy policy'`	Adds a description for this power supply redundancy policy. If the text includes spaces, it must be enclosed in single quotes.
Step 4	[**no**] **redundancy-mode** {**combined** | **ps-redundant** | **redundant**} **Example:** `apic1(config-power)# redundancy-mode ps-redundant`	Specifies power supply redundancy mode. • **combined** — This mode does not provide power redundancy. The available power is the total power capacity of all power supplies. • **ps-redundant** —This mode provides an extra power supply in case an active power supply goes down. The power supply that can supply the most power operates in standby mode. The other one or two power supplies are active. The available power is the amount of power provided by the active power supply units. • **redundant** —This mode combines power supply redundancy and input source redundancy, which means that the chassis has an extra power supply and each half of each power supply is connected to one electrical grid while the other half of each power supply is connected to the other electrical grid. The available power is the lesser of the available power for power supply mode and input source mode.

Examples

This example shows how to configure a power supply redundancy policy for the ps-redundant mode.

```
apic1# configure
apic1(config)# power redundancy-policy myPowerPolicy
apic1(config-pod)# isis fabric
apic1(config-power)# description 'This is my power redundancy policy'
apic1(config-power)# redundancy-mode ps-redundant
```

Configuring a Scheduler

A schedule allows operations, such as configuration import/export or tech support collection, to occur during one or more specified windows of time.

A schedule contains a set of time windows (occurrences). These windows can be one time only or can recur at a specified time and day each week. The options defined in the window, such as the duration or the maximum number of tasks to be run, determine when a scheduled task will execute. For example, if a change cannot be deployed during a given maintenance window because the maximum duration or number of tasks has been reached, that deployment is carried over to the next maintenance window.

Each schedule checks periodically to see whether the APIC has entered one or more maintenance windows. If it has, the schedule executes the deployments that are eligible according to the constraints specified in the maintenance policy.

A schedule contains one or more occurrences, which determine the maintenance windows associated with that schedule. An occurrence can be one of the following:

- Absolute (One Time) Window—An absolute window defines a schedule that will occur only once. This window continues until the maximum duration of the window or the maximum number of tasks that can be run in the window has been reached.
- Recurring Window—A recurring window defines a repeating schedule. This window continues until the maximum number of tasks or the end of the day specified in the window has been reached.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **scheduler** *schedule-name* **Example:** `apic1(config)# scheduler controller schedule myScheduler`	Creates a new scheduler or configures an existing scheduler.
Step 3	[**no**] **description** *text* **Example:** `apic1(config-scheduler)# description 'This is my scheduler'`	Adds a description for this scheduler. If the text includes spaces, it must be enclosed in single quotes.
Step 4	[**no**] **absolute window** *window-name* **Example:** `apic1(config-scheduler)# absolute window myAbsoluteWindow`	Creates an absolute (one time) window schedule.

	Command or Action	Purpose
Step 5	[**no**] **max concurrent nodes** *count* **Example:** `apic1(config-scheduler-absolute)# max concurrent nodes 300`	Sets the maximum number of nodes (tasks) that can be processed concurrently. The range is 0 to 65535. Set to 0 for unlimited nodes.
Step 6	[**no**] **max running time** *time* **Example:** `apic1(config-scheduler-absolute)# max running time 00:01:30:00`	Sets the maximum running time for tasks in the format dd:hh:mm:ss. The range is 0 to 65535. Set to 0 for no time limit.
Step 7	[**no**] **time start** *time* **Example:** `apic1(config-scheduler-absolute)# time start 2016:jan:01:12:01`	Sets the starting time in the format [[[yyyy:]mmm:]dd:]HH:MM.
Step 8	**exit** **Example:** `apic1(config-scheduler-absolute)# exit`	Returns to scheduler configuration mode.
Step 9	[**no**] **recurring window** *window-name* **Example:** `apic1(config-scheduler)# recurring window myRecurringWindow`	Creates a recurring window schedule.
Step 10	[**no**] **max concurrent nodes** *count* **Example:** `apic1(config-scheduler-recurring)# max concurrent nodes 300`	Sets the maximum number of nodes (tasks) that can be processed concurrently. The range is 0 to 65535. Set to 0 for unlimited nodes.
Step 11	[**no**] **max running time** *time* **Example:** `apic1(config-scheduler-recurring)# max running time 00:01:30:00`	Sets the maximum running time for tasks in the format dd:hh:mm:ss. The range is 0 to 65535. Set to 0 for no time limit.
Step 12	[**no**] **time start** {**daily** *HH:MM* | **weekly** (See usage) *HH:MM*} **Example:** `apic1(config-scheduler-recurring)# time start weekly wednesday 12:30`	Sets the period (daily or weekly) and starting time. If **weekly** is selected, choose from these options: • **monday** • **tuesday** • **wednesday** • **thursday** • **friday** • **saturday**

	Command or Action	Purpose
		• **sunday** • **even-day** • **odd-day** • **every-day**

Examples

This example shows how to configure a recurring scheduler to run every Wednesday.

```
apic1# configure
apic1(config)# scheduler controller schedule myScheduler
apic1(config-scheduler)# description 'This is my scheduler'
apic1(config-scheduler)# recurring window myRecurringWindow
apic1(config-scheduler-recurring)# max concurrent nodes 300
apic1(config-scheduler-recurring)# max running time 00:01:30:00
apic1(config-scheduler-recurring)# time start weekly wednesday 12:30
```

Configuring System MTU

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	[**no**] **system jumbomtu** *size* **Example:** apic1(config)# **`system jumbomtu 9000`**	Sets the maximum transmit unit (MTU) for host facing ports. Up to Cisco APIC Release 3.1(2), the range is 576 to 9000 bytes. From release 3.1(2), and later, the maximum MTU value is 9216. The default has not changed from 9000.

Examples

This example shows how to configure the system MTU size.

```
apic1# configure terminal
apic1(config)# system jumbomtu 9000
```

About PTP

Precision Time Protocol (PTP) is a time synchronization protocol defined in IEEE 1588 for nodes distributed across a network. With PTP, it is possible to synchronize distributed clocks with an accuracy of less than 1 microsecond via Ethernet networks. PTP's accuracy comes from the hardware support for PTP in the ACI fabric spines and leafs. It allows the protocol to accurately compensate for message delays and variation across the network.

PTP is a distributed protocol that specifies how real-time PTP clocks in the system synchronize with each other. These clocks are organized into a master-slave synchronization hierarchy with the grandmaster clock, which is the clock at the top of the hierarchy, determining the reference time for the entire system. Synchronization is achieved by exchanging PTP timing messages, with the members using the timing information to adjust their clocks to the time of their master in the hierarchy. PTP operates within a logical scope called a PTP domain.

The PTP process consists of two phases: establishing the master-slave hierarchy and synchronizing the clocks. Within a PTP domain, each port of an ordinary or boundary clock follows this process to determine its state:

- Examines the contents of all received announce messages (issued by ports in the master state).
- Compares the data sets of the foreign master (in the announce message) and the local clock for priority, clock class, accuracy, and so on.
- Determines its own state as either master or slave.

After the master-slave hierarchy has been established, the clocks are synchronized as follows:

- The master sends a synchronization message to the slave and notes the time it was sent.
- The slave receives the synchronization message and notes the time that it was received. For every synchronization message, there is a follow-up message. Hence, the number of sync messages should be equal to the number of follow-up messages.
- The slave sends a delay-request message to the master and notes the time it was sent.
- The master receives the delay-request message and notes the time it was received.
- The master sends a delay-response message to the slave. The number of delay request messages should be equal to the number of delay response messages.
- The slave uses these timestamps to adjust its clock to the time of its master.

In ACI fabric, when PTP feature is globally enabled in APIC, the software automatically enables PTP on specific interfaces of all the supported spines and leafs. This auto-configuration ensures that PTP is optimally enabled on all the supported nodes. In the absence of an external grandmaster clock, one of the spine switch is chosen as the grandmaster. The master spine is given a different PTP priority as compared to the other spines and leaf switches so that they will act as PTP slaves. This way we ensure that all the leaf switches in the fabric synchronize to the PTP clock of the master spine.

If an external Grandmaster clock is connected to the spines, the spine syncs to the external GM and in turn acts as a master to the leaf nodes.

PTP Default Settings

The following table lists the default settings for PTP parameters.

Parameters	Default
PTP device type	Boundary clock
PTP clock type	Two-step clock
PTP domain	0
PTP priority 1 value when advertising the clock	255
PTP priority 2 value when advertising the clock	255
PTP announce interval	1 log second
PTP announce timeout	3 announce intervals
PTP delay-request interval	0 log seconds
PTP sync interval	-2 log seconds
PTP VLAN	1

Note PTP operates only in boundary clock mode. Cisco recommends deployment of a Grand Master Clock (10 MHz) upstream, with servers containing clocks requiring synchronization connected to the switch.

PTP Verification

Command	Purpose
show ptp brief	Displays the PTP status.
show ptp clock	Displays the properties of the local clock, including clock identity.
show ptp clock foreign-masters record interface ethernet *slot/port*	Displays the state of foreign masters known to the PTP process. For each foreign master, the output displays the clock identity, basic clock properties, and whether the clock is being used as a grandmaster.
show ptp corrections	Displays the last few PTP corrections.
show ptp counters [**all** |**interface Ethernet** *slot/port*]	Displays the PTP packet counters for all interfaces or for a specified interface.
show ptp parent	Displays the properties of the PTP parent.

Guidelines and Limitations

Follow these guidelines and limitations:

- Latency requires all the nodes in the fabric to be synchronized using Precision Time Protocol (PTP).

- Latency measurement and PTP are only supported on the following switches:
  - N9K-C93108TC-EX
  - N9K-C93108TC-FX
  - N9K-C93180LC-EX
  - N9K-C93180YC-EX
  - N9K-C93180YC-FX
  - N9K-C9364C
  - N9K-X9732C-EX
  - N9K-X9736C-EX
  - N9K-X9736C-FX
- Latency measurement is supported only for the packets that ingress, egress, and transit through EX or FX-based TORs.
- All the spine nodes in the fabric should have EX or FX-based line cards to support PTP.
- PTP and the latency feature is not supported on any N9K-C93128TX, N9K-C9396PX, and N9K-C9396TX TORs or spine switches. In the presence of non-EX/FX TORs in the fabric, we recommend that you have the external GM connectivity to all the spine switches to ensure that the PTP time is synced across all the supported TORs.
- External Grandmaster (GM) clock is not mandatory for PTP in a single Pod. If there is no external GM connected to the ACI fabric, one of the spine nodes acts as the GM. This spine switch has a PTP priority1 value as 254. All the other spine switches and leaf switches in the fabric will synchronize their clock to this Master spine switch clock. If the external GM is connected later to the spine switch, it should have a priority value less than 254 for it to act as the GM for the entire fabric.
- External Grandmaster clock is mandatory for PTP in a multipod scenario. In addition, external GM needs to be connected to the IPN such that the Grandmaster clock is the master to the spine switches in different PODs. The spine switches connected to IPN will act as the boundary clock and all the nodes within the POD will sync their clock this spine switch.
- PTP operates only in boundary clock mode. End-to-end transparent clock and peer-to-peer transparent clock modes are not supported.
- PTP supports transport over User Datagram Protocol (UDP). Transport over Ethernet is not supported.
- PTP supports multicast communication only; unicast mode is not supported.
- Beginning with release 4.0(1), support is added for changing the resolution factor to 11 which then can measure up to 214 milliseconds with an accuracy of 204ns.

Configuring PTP Using the NX-OS CLI

Procedure

Step 1 Enable PTP.

Example:

```
Enable ptp:
========
apic# configure terminal
apic(config)# ptp
Disable ptp:
========
apic# configure terminal
apic(config)# no ptp
```

Step 2 To verify PTP on ACI switches:

Example:

```
leaf1# show ptp brief
PTP port status
-----------------------
Port         State
-------  --------------
Eth1/49     Slave

leaf1#
leaf1#
leaf1# show ptp clock
PTP Device Type: Boundary clock
Clock Identity :  0c:75:bd:ff:fe:03:1d:10
Clock Domain: 0
Number of PTP ports: 1
Priority1 : 255
Priority2 : 255
Clock Quality:
        Class : 248
        Accuracy : 254
        Offset (log variance) : 65535
Offset From Master : 32
Mean Path Delay : 128
Steps removed : 1
Local clock time:Thu Jul 27 19:43:42 2017

leaf1#
leaf1# show ptp clock foreign-masters record interface ethernet 1/49

P1=Priority1, P2=Priority2, C=Class, A=Accuracy,
OSLV=Offset-Scaled-Log-Variance, SR=Steps-Removed
GM=Is grandmaster

---------   -----------------------  ---  ----  ----  ---  -----  --------
Interface         Clock-ID           P1   P2    C     A    OSLV   SR
---------   -----------------------  ---  ----  ----  ---  -----  --------

Eth1/49    d4:6d:50:ff:fe:e6:4d:3f   254  255   248   254  65535  0    GM

leaf1#
```

```
leaf1#
leaf1# show ptp corrections

PTP past corrections
-------------------------------------------------------------------------------------
Slave Port              SUP Time                Correction(ns)    MeanPath Delay(ns)
----------  ------------------------------  ------------------  ------------------
Eth1/49       Thu Jul 27 19:44:11 2017 364281    36                  152
Eth1/49       Thu Jul 27 19:44:11 2017 114565    16                  132
Eth1/49       Thu Jul 27 19:44:10 2017 862912    8                   132
Eth1/49       Thu Jul 27 19:44:10 2017 610823    8                   132
Eth1/49       Thu Jul 27 19:44:10 2017 359557    16                  132
Eth1/49       Thu Jul 27 19:44:10 2017 109937    8                   132
Eth1/49       Thu Jul 27 19:44:09 2017 858113    16                  132
Eth1/49       Thu Jul 27 19:44:09 2017 606536    16                  132
Eth1/49       Thu Jul 27 19:44:09 2017 354837    -16                 132
Eth1/49       Thu Jul 27 19:44:09 2017 104226    24                  148
Eth1/49       Thu Jul 27 19:44:08 2017 853263    24                  148
Eth1/49       Thu Jul 27 19:44:08 2017 601780    16                  148
Eth1/49       Thu Jul 27 19:44:08 2017 349639    -4                  148
Eth1/49       Thu Jul 27 19:44:08 2017 99970     16                  144
Eth1/49       Thu Jul 27 19:44:07 2017 848507    0                   144
Eth1/49       Thu Jul 27 19:44:07 2017 596143    24                  144
Eth1/49       Thu Jul 27 19:44:07 2017 344808    4                   144
Eth1/49       Thu Jul 27 19:44:07 2017 93156     -16                 140
Eth1/49       Thu Jul 27 19:44:06 2017 843263    24                  140
Eth1/49       Thu Jul 27 19:44:06 2017 590189    8                   140
leaf1#
leaf1#
leaf1# show ptp counters all

PTP Packet Counters of Interface Eth1/49:
----------------------------------------------------------------
Packet Type                 TX                      RX
----------------    --------------------    --------------------
Announce                          56                    5424
Sync                             441                   43322
FollowUp                         441                   43321
Delay Request                   7002                       0
Delay Response                     0                    7002
PDelay Request                     0                       0
PDelay Response                    0                       0
PDelay Followup                    0                       0
Management                         0                       0
----------------------------------------------------------------

leaf1#
leaf1#
leaf1# show ptp parent

PTP PARENT PROPERTIES

Parent Clock:
Parent Clock Identity:  d4:6d:50:ff:fe:e6:4d:3f
Parent Port Number: 258
Observed Parent Offset (log variance): N/A
Observed Parent Clock Phase Change Rate: N/A

Grandmaster Clock:
Grandmaster Clock Identity:  d4:6d:50:ff:fe:e6:4d:3f
Grandmaster Clock Quality:
        Class: 248
        Accuracy: 254
        Offset (log variance): 65535
```

```
        Priority1: 254
        Priority2: 255

leaf1#
```

Step 3 To verify troubleshooting steps:

Example:

```
apic1# show troubleshoot eptoep session eptoep latency

Source --> Destination
Last Collection(30 seconds)
+--------------------+------------------------------+---------------+
| Average (microsec) | Standard Deviation (microsec) | Packet Count |
+--------------------+------------------------------+---------------+
| 18                 | 24                           | 1086         |
|                    |                              |              |
+--------------------+------------------------------+---------------+



Cumulative
+--------------------+----------------+--------------------+
| Average (microsec) | Max (microsec) | Total Packet Count |
+--------------------+----------------+--------------------+
| 18                 | 202            | 6117438            |
|                    |                |                    |
+--------------------+----------------+--------------------+
```