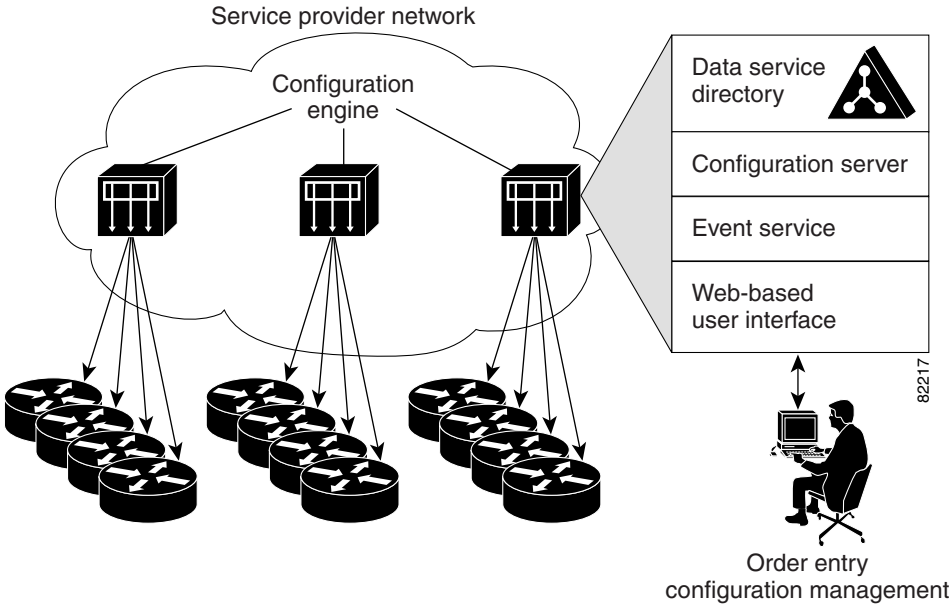
Product Overview

This chapter provides a high-level overview of the Cisco CNS Configuration Engine. It is organized as follows:

The Cisco CNS Configuration Engine is a network management application that acts as a configuration service for automating the deployment and management of network devices and services (see Figure 1-1). The Cisco CNS Configuration Engine runs on the Cisco CNS 2100 Series Intelligence Engine (CNS 2100 Series system) hardware platform.

Figure 1-1 Cisco CNS Configuration Engine Architectural Overview



Each Cisco CNS Configuration Engine manages a group of Cisco IOS devices (routers) and services they deliver, storing their configurations and delivering them as needed. The Cisco CNS Configuration Engine automates initial configurations and configuration updates by generating device-specific configuration changes, sends them to the device, executes the configuration change, and logs the results.

The Cisco CNS Configuration Engine is a web-based system for automatically distributing configuration files to Cisco IOS network devices running Cisco IOS version 12.2(2) T, or later where authentication is NOT used. Cisco IOS version 12.2(11)T is required if encryption or authentication is used.

Note If you are running devices that use an earlier version of Cisco IOS, or a different operating system, such as Catalyst, you should invoke the Intelligent Modular Gateway for communicating with the device. For more information about Intelligent Modular Gateway, see "Intelligent Modular Gateway" section on page 1-5.

The Cisco CNS Configuration Engine utilizes the following popular industry standards and technologies:

- eXtensible Markup Language (XML)
- Java naming directory interface (JNDI)
- Hypertext Transport Protocol (HTTP)
- Java servlets
- Lightweight Directory Access Protocol (LDAP)

The Cisco CNS Configuration Engine supports two modes of operation (Internal Directory and External Directory) and it includes the following Cisco Networking Services (CNS) components:

- Configuration service (web server, file manager, and namespace mapping server)
- Event service (event gateway)
- Data service directory (data models and schema)
- Intelligent Modular Gateway (IMGW)

The Cisco CNS Configuration Engine can be used as the runtime component for deployment of customer-developed applications. These applications can be developed using the Cisco CNS SDK 1.5.

Modes of Operation

There are two modes of system operation for the Cisco CNS Configuration Engine:

- Internal Directory Mode
- External Directory Mode

Internal Directory Mode

In Internal Directory mode, the Cisco CNS Configuration Engine supports an embedded CNS Directory Service. In this mode, no external directory or other data store is required. To store device configuration information, the Cisco CNS Configuration Engine uses the CNS data models implemented as an extended X.500 directory schema in the CNS Directory Service.

External Directory Mode

In External Directory mode, the Cisco CNS Configuration Engine supports the use of a user-defined external directory. In this mode, the Cisco CNS Configuration Engine supports the following directory services:

- Novell Directory Services
- Microsoft Active Directory
- iPlanet

CNS Configuration Service

The CNS Configuration Service is the core component of the Cisco CNS Configuration Engine. It consists of a configuration server that works in conjunction with configuration agents located at each router. The CNS Configuration Service delivers device and service configurations to Cisco IOS devices for initial configuration and mass reconfiguration by logical groups. Routers receive their initial configuration from the CNS Configuration Service when they start up on the network the first time.

The CNS Configuration Service uses the CNS Event Service to send and receive events required to apply configuration changes and send success and failure notifications.

The configuration server consists of a web server that uses configuration templates and the device-specific configuration information stored in the embedded (Internal Directory mode) or remote (External Directory mode) directory.

Configuration templates are text files containing static configuration information in the form of command-line interface (CLI) commands. In the templates, variables are specified using lightweight directory access protocol (LDAP) URLs that reference the device-specific configuration information stored in a directory.

The configuration template includes additional features that allow simple conditional control structures and modular sub-templates in the configuration template (see the "Templates and Template Management" section on page 3-42).

The configuration server uses Hypertext Transport Protocol (HTTP) to communicate with the CNS Configuration Agent running on the managed Cisco IOS device. The configuration server transfers data in eXtensible Markup Language (XML) format. The configuration agent in the router uses its own XML parser to interpret the configuration data and remove the XML tags from the received configuration.

The configuration agent can also perform a syntax check on received configuration files. The configuration agent can also publish events through the event gateway to indicate the success or failure of the syntax check.

The configuration agent can either apply configurations immediately or delay the application until receipt of a synchronization event from the configuration server.

CNS Event Service

The Cisco CNS Configuration Engine uses the CNS Event Service for receipt and generation of configuration events. The CNS Event Agent resides on Cisco IOS devices and facilitates communication between routers and the event gateway on the Cisco CNS Configuration Engine.

The CNS Event Service is a highly-scalable publish and subscribe communication method. The CNS Event Service uses subject-based addressing to help messages reach their destination. Subject-based addressing conventions define a simple, uniform namespace for messages and their destinations.

NameSpace Mapper

The Cisco CNS Configuration Engine includes the NameSpace Mapper (NSM) that provides a lookup service for managing logical groups of devices based on application, device/group ID, and event.

For a subscriber, when given a unique device ID and event, the namespace mapping service returns a set of events to which to subscribe. Similarly, for a publisher, when given a unique group ID, device ID, and event, the mapping service returns a set of events on which to publish.

The CNS Namespace Mapping Service allows you to address multiple network devices by a single posting of a publish or subscribe event, and it allows your network administrator to map Cisco-standardized event names to names of his or her choosing.

For example, in a network of 100 routers, there may be 10 which the administrator wants to configure as a VPN (Virtual Private Network). In order to load a configuration into each of these devices, your client application could either publish 10 *cisco.cns.config.load* events, or the administrator could associate the 10 devices with a common group name and your client application can post the event once. The administrator could rename the *cisco.cns.config.load* subject to *application.load* and group all the devices in the West Coast under a group called “westcoast.” Then the application would just have to publish on *application.load.westcoast* and the devices in the “westcoast” group would get the event.

NSM Modes

The NameSpace Mappers can operate in one of three NSM modes:

- Default
- Provider
- None

The NSM mode is set when you run the **Setup** program (see “Running the Setup Program” section on page 2-1).

Default Mode

No directory setup is required for Default mode. The DeviceID is just appended to the subject. This allows you to individually address a device.

To set Default mode, use **default** for the value of the NSM Directive parameter in the **Setup** program.

Provider Mode

Directory setup is required for Provider mode. NSM looks up the directory for subject mappings for a device. This mode allows you to address a group of devices in one event.

To set Provider mode, use **http** for the value of the NSM Directive parameter in the **Setup** program.

None Mode

No directory setup is required for None mode. No subject mapping is done. All devices are subscribed to the same subject and respond on the same subject.

Note This mode should be used only for broadcast events.

To set None mode, use **none** for the value of the NSM Directive parameter in the **Setup** program.

More information about NSM can be found in the *CNS SDK 1.5 Programmer's Guide and API Reference*.

Directory setup can be done using the Directory Administration Tool (see "Directory Administration Tool" section on page 5-1.

Event Gateway

The CNS Event Gateway acts as a relay between the CNS Integration Bus and CNS agent-enabled devices, which enables event-based communication.

The CNS Event Gateway uses NSM to map subjects. The mode of operation is determined by the value set for the NSM Directive parameter during **Setup**.

If you choose the Provider mode (**http**), the Event Gateway must be given a parameter that indicates which application namespace must be used for subject mapping. The Cisco CNS Configuration Engine prompts for this parameters value during **Setup** with the message:

```
Enter NSM directive (none, default, http):
```

The default value for this parameter is **default**. However, during **Setup**, you can override this value with one of your own.

Note If you are migrating groups and devices from Release 1.2 to 1.3, you must use the value for this parameter when you establish reference to an application namespace for this NSM mode. For more information, see "Importing Groups and Devices from Release 1.2 to 1.3" section on page 2-22.

Each Event Gateway process can support up to a maximum of 500 devices. To support more than 500 devices, you can run multiple gateway processes. During **Setup**, you can set the number of concurrent gateway processes to start with either one or both of the following prompts, depending on how you want to setup your SSL (see "Encryption" section on page 1-9) communications:

```
Enter number of Event Gateways that will be started with crypto operation:
Enter number of Event Gateways that will be started with plaintext operation:
```

Intelligent Modular Gateway

Intelligent Modular Gateway allows you to run the Cisco CNS Configuration Engine for automatically distributing configuration files to Cisco IOS network devices running Cisco IOS versions earlier than 12.2(2) T; as well as to Catalyst switches, CCS 11k devices, Cache Engines, and PIX firewalls.

Note If you are running devices that use Cisco IOS version 12.2(2) T or later, you should use the CNS Event Gateway.

The Intelligent Modular Gateway accomplishes this task by adding the ability to use alternate access methods to connect to devices that do not have CNS agents in their software. Currently, the access methods are Telnet or SSH.

The interface to the Intelligent Modular Gateway is the same as that of the CNS Event Gateway. It responds to the same events. The NameSpace Mapper operates in the same way. Therefore, once some initial setup work is done, applications need not know the difference between communicating with agent-enabled devices by way of the Event Gateway and non-agent devices by way of the Intelligent Modular Gateway.

Restrictions

Using the Intelligent Modular Gateway with a Telnet or SSH transport creates some restrictions in terms of how the Cisco CNS Configuration Engine architecture is used.

- When using Telnet or SSH as a transport, no syntax checking can be done on the configurations before they are applied.

  Syntax checking in the Cisco CNS Configuration Engine architecture is accomplished by an intelligent agent in the device that has access to internal parser functions. A Telnet or SSH interface does not provide any means to access this functionality. Therefore, any syntax checking attributes are ignored. Errors are only detected when the configuration is actually applied and applications must deal with the fact that configuration lines prior to the error were executed.

- Because all logic is external to the device, there is no way to watch for configuration changes that are done outside the scope of the network management software.

  For example, if a network administrator uses a standard Telnet or SSH client to directly access a network element and changes the configuration, that element would not be synchronized with the network management infrastructure, and depending on the change, might become unmanageable. This is especially true if the login mechanisms (usernames and passwords) are changed. Login mechanism changes should be handled during a maintenance window, during which event-based configuration is not occurring, so that race conditions do not occur. Any such changes must be reflected on the provisioning system's device information screen so that the Device Information Database is properly updated before any new partial configurations are sent.

- The scope of error checking upon configuration load is limited to syntax checking.

  Semantic errors cannot be detected. The output is returned in a buffer that applications should log. In a case where something is not operating properly, a network administrator can manually look at the log of what the device was reporting and determine if a semantic error occurred.

- The initial configuration mechanism as defined in the Cisco CNS Configuration Engine architecture is not supported.

  This mechanism allows a router to be preconfigured with the **cns config initial** command, causing it to contact the configuration server to retrieve its initial configuration. However, because the legacy devices do not have the agent code in them, they can never contact the configuration server (they do not understand the configuration command). Therefore, this mechanism does not make sense when using Telnet or SSH as a transport. If an initial configuration needs to be delivered by the Cisco CNS Configuration Engine, it has to be done through the partial configuration mechanism.

- Aside from the device information database, the gateway is stateless.

There is no read back of configurations to make sure they were applied, nor is there automatic rollback of configurations if a failure occurs.

- If a device is not directly connected to the management network, it must be attached through a Cisco 2511 communication server.

  The API allows you to set up an arbitrary network topology to reach the device. However, this release only supports two possible topologies: direct connection to one of the device network interfaces, or console access by way of a Cisco access server, such as a 2511.

- Device failures are only detected within a user-specified polling interval.

  This is because while the standard Event Gateway requires that routers maintain a connection to the Event Gateway (so any breakage of that connection would signal a problem), the Telnet or SSH interface is implemented through a transient connection. Therefore, the gateway must poll all devices at some user-specified interval to make sure they are responding, so failure detection is not immediate.

- When both agent-enabled and legacy devices are present on the same network, it is recommended that both gateways be run at the same time.

  The standard (CNS) Event Gateway talks to the agent-enabled devices and the Intelligent Modular Gateway talks to the legacy devices.

Note Do not put an entry in the Device Information Database for a router that is already agent-enabled because both gateways will try to control the router and unpredictable results may occur.

Modular Router Support

The template mechanism for the devices has been enhanced to support modular routers. A modular router chassis includes slots in which you can install modules. For example, the Cisco 3660 (see Figure 1-2) has six network module slots. You can install any module into any available slot in the chassis. Some modules like 2 Ethernet 2 WAN card slot module can in turn have sub slots to install interface cards or line cards (see Figure 1-3 on page 1-8). Device management has been extended to support sub-devices representing line cards.

Figure 1-2 Cisco 3660 Modular Router

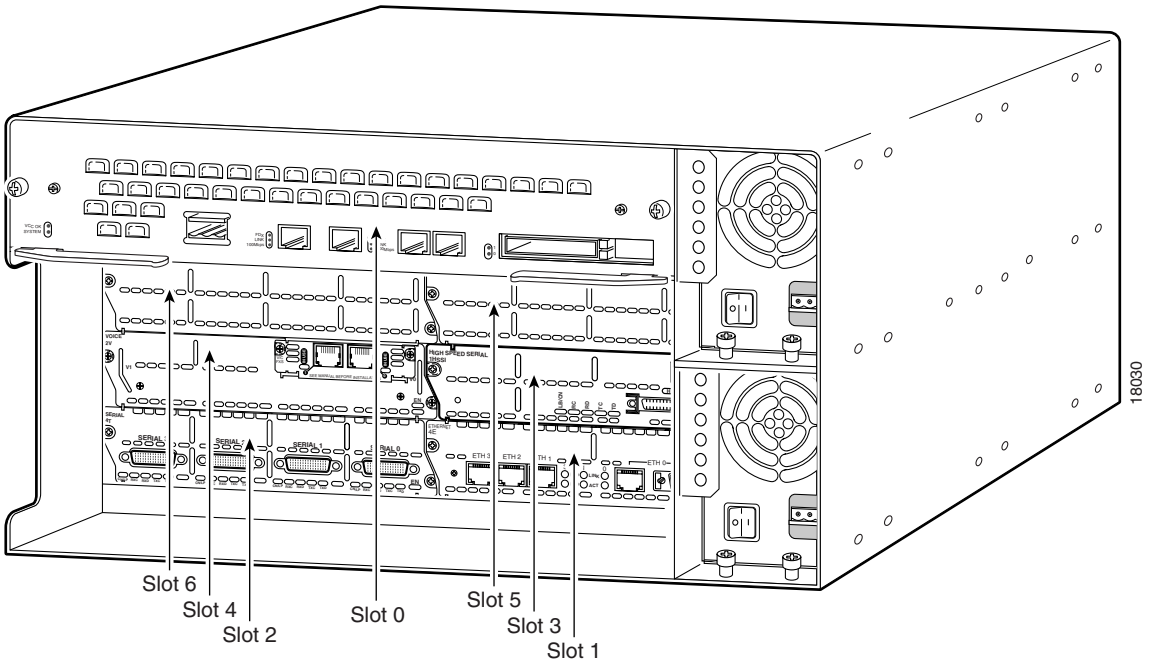
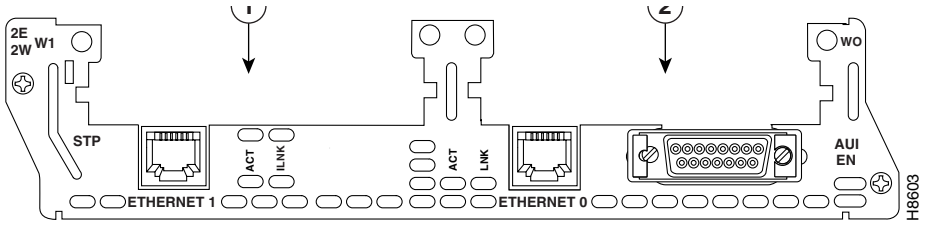


Figure 1-3 Interface or Line Card Slots



Additional attributes representing line card type and sub-devices have been added to the existing device object structure in the directory server in order to have the same structure to represent the main device or the subdevice.

For a modular router, a subdevice configuration object and configuration template is defined for every network module whose interfaces need to be configured and for which the interface number can be variable; based on the slot. Then, a device configuration object and a template is defined for the main device. Fixed interface numbers can be configured in the main device template.

Modular router events are published to the event bus and are accessible to applications connected to the bus. The Cisco IOS device publishes the system hardware configuration in the *cisco.cns.config.device-details* event after hardware discovery. The Cisco CNS Configuration Engine is configured to listen for this event, retrieve it and extract the hardware configuration of the device.

In Internal Directory mode, modular router support sessions work with NSM in all three modes (see "NSM Modes" section on page 1-4).

Data Administration Tool

The Data Administration Tool (DAT) presents you with a web-based user interface that allows you to populate and manage the data in the directories. You can View/Add/Delete/Update devices (CNS agent-enabled devices, see "Intelligent Modular Gateway" section on page 1-5), groups of devices, and applications in the directory. Also, you can View/Add/Delete/Update events specific to each application.

DAT also provides you with the additional capability of bulk data upload.

Note You cannot change (extend) the schema using DAT. You have to populate the schema manually in the directory server.

For information about how to use DAT, see "Directory Administration Tool" section on page 5-1.

Encryption

Secure Socket Layer (SSL) method has been adopted as the encryption mechanism for HTTP sessions between the configuration agent and the configuration server, and the TCP session between the CNS Event Gateway and the event agent.

To use encryption, the Cisco IOS devices must be running a crypto image and version 12.2(11)T of the Cisco IOS.

Device Authentication

The configuration server and CNS Event Gateway are supplied with a X.509 certificate generated by a certificate authority (CA) server. It is responsibility of the network administrator to have a CA server and to control certificate generation and revocation.

The Cisco IOS device must to be configured to recognize the CA. There is no client side certificate in the Cisco IOS device.

For the configuration server, once the Cisco IOS device has validated the certificate, it sends {hostname:cns_password} over the encrypted pipe. The device uses a CNS password to be authenticated by the Cisco CNS Configuration Engine.

Note Authentication is also done when the links are in clear text.

A server configured for secure connections is also able to enact non-secure (clear-text) sessions. The password check is done regardless of whether encryption is used or not.

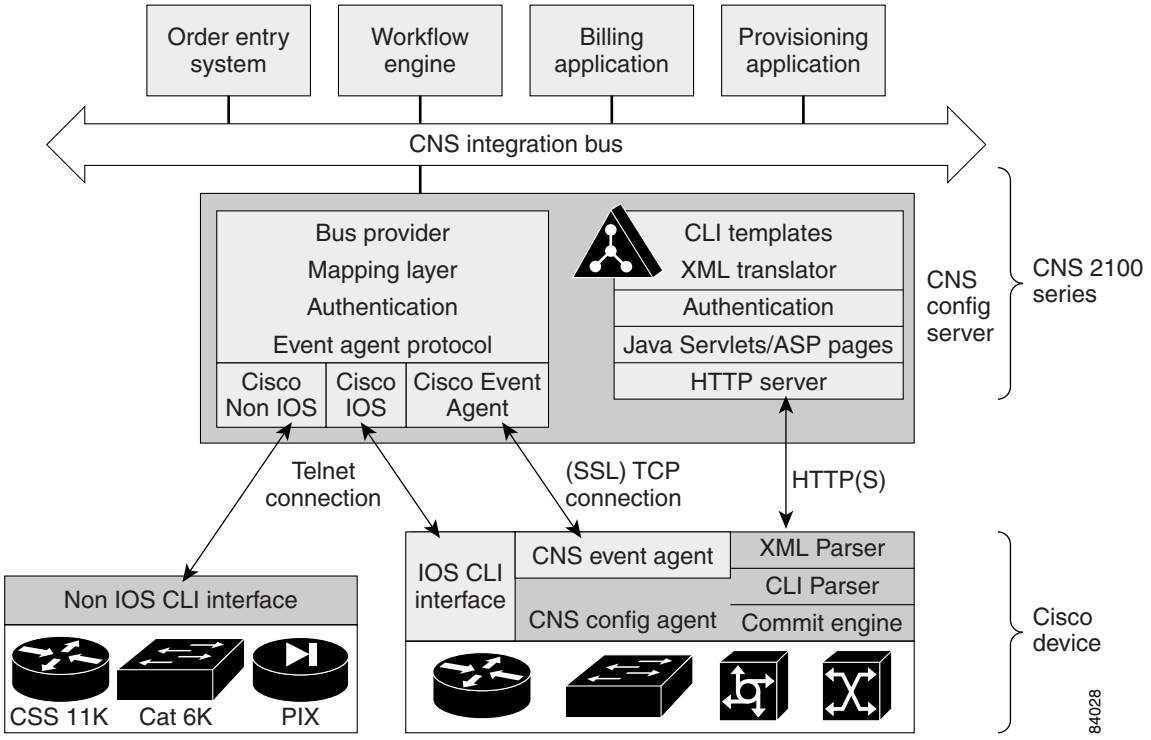
Once the server is secured, it is no longer be able to process requests that do not have a password. It cannot tell the difference between a clear-text request from a device in a secure environment from a device in an non-secure environment.

For the CNS Event Gateway, once the Cisco IOS device has validated the certificate, it sends a DeviceID control message over the encrypted pipe that has the CNS password of the device. The {hostname:cns_password} is validated using the authentication API. If it is not matched, the SSL session is terminated and an entry made to the security log. This ensures only authorized customer premises equipment (CPE) devices connect to the CNS Event Gateway and are able to use the CNS Integration Bus.

How the Cisco CNS Configuration Engine Works

The Cisco CNS Configuration Engine dynamically generates Cisco IOS configuration files (documents), packages these file in XML format, and distributes them by means of Web/HTTP (see Figure 1-4 on page 1-10). This takes place in response to a *pull* (get) operation.

Figure 1-4 Configuration Registrar Functional Diagram

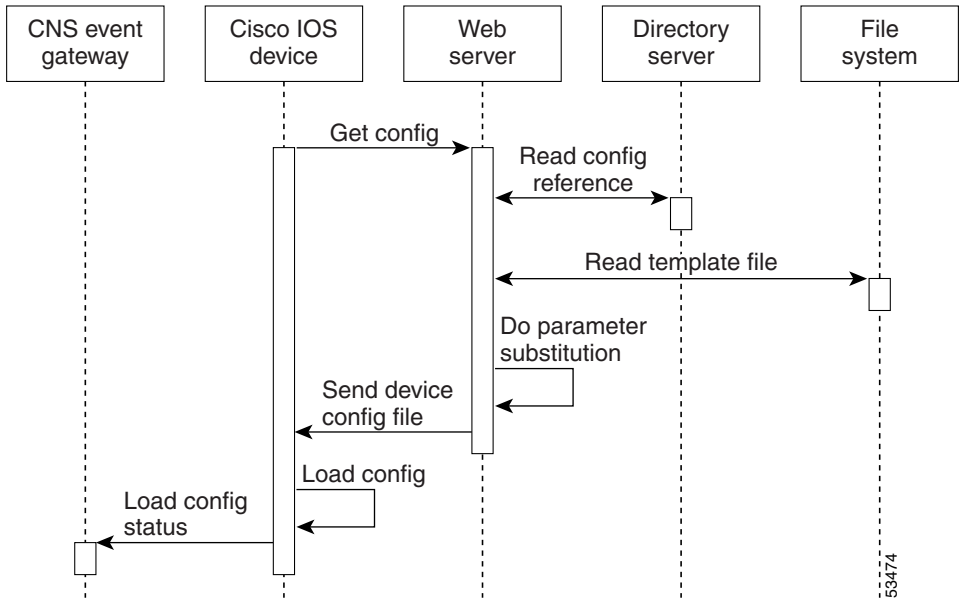


A Cisco IOS device initiates a get operation when it first appears on the network (**cns config init…**) or when notified (by subscribed event) of a configuration update (**cns config partial…**).

Note For more information about these and other related CLI commands, refer to the Cisco IOS configuration guide and command reference publications.

When a Cisco IOS device issues a request for a device configuration file, the request includes a unique identifier (configID = hostname) used to help locate the relevant configuration file parameters for this device on the directory server. Figure 1-5 shows the process flow for a configuration load operation.

Figure 1-5 Configuration Load Process Flow



When the web server receives a request for a configuration file, it invokes the Java Servlet and executes the embedded code. This directs the web server to access the directory server and file system to read the configuration reference for this device and template. The configuration server prepares an instantiated configuration file by substituting all the parameter values specified in the template with valid values for this device. The configuration server forwards the configuration file to the web server for transmission to the Cisco IOS device.

The configuration agent at the router accepts the configuration file from the web server, performs XML parsing, syntax checking (optional), and loads the configuration file. The router reports the status of the configuration load as an event that can be subscribed to by a network monitoring or workflow application.

Load Initial Configuration

1. The Cisco CNS Configuration Engine reads the template files.
2. The Cisco CNS Configuration Engine does the parameter substitution.
3. The Cisco CNS Configuration Engine sends the device configuration to the Cisco IOS device.
4. The Cisco IOS device tries to load the initial configuration.
5. The Cisco IOS device publishes the load configuration status event to the event gateway.

Modular Router

1. The modular router posts an HTTP request containing the hardware configuration to the Cisco CNS Configuration Engine for the initial configuration.
2. The Cisco CNS Configuration Engine reads the hardware configuration of the device from the HTTP request and updates the directory server with the latest configuration.
3. The Cisco CNS Configuration Engine reads the template files.
4. The Cisco CNS Configuration Engine does the parameter substitution.

5. The Cisco CNS Configuration Engine sends the device configuration to the Cisco IOS device.
6. The modular router tries to load the initial configuration.
7. The modular router publishes the load configuration status event to the event gateway.

Load Partial Configuration

1. The user modifies a template in the Cisco CNS Configuration Engine user interface.
2. The template contents are passed to the Cisco CNS Configuration Engine.
3. The Cisco CNS Configuration Engine stores the template in the file system.
4. The user clicks the update device button in the user interface.
5. The Cisco CNS Configuration Engine publishes a *cisco.cns.config.load* event.
6. The Cisco IOS device retrieves the *cisco.cns.config.load* event.
7. The Cisco CNS Configuration Engine reads the template files.
8. The Cisco CNS Configuration Engine does the parameter substitution.
9. The Cisco CNS Configuration Engine sends the device configuration to the Cisco IOS device.
10. The Cisco IOS device tries to load the partial configuration.
11. The Cisco IOS device publishes the load configuration status event to the event gateway.

Modular Router

1. The user modifies a template in the Cisco CNS Configuration Engine user interface.
2. The template contents are passed to the Cisco CNS Configuration Engine.
3. The Cisco CNS Configuration Engine stores the template in the file system.
4. The user clicks the update device button in the user interface.
5. The Cisco CNS Configuration Engine publishes a *cisco.cns.config.load* event.
6. The modular router retrieves the *cisco.cns.config.load* event.
7. The Cisco IOS device posts a HTTP request containing the hardware configuration to the Cisco CNS Configuration Engine for the partial configuration.
8. The Cisco CNS Configuration Engine reads the template files.
9. The Cisco CNS Configuration Engine does the parameter substitution.
10. The Cisco CNS Configuration Engine sends the device configuration to the modular router.
11. The modular router tries to load the partial configuration.
12. The modular router publishes the load configuration status event to the event gateway.

How EventID, and ConfigID are Used

The Cisco CNS Configuration Engine intersects two name spaces, one for the event bus and the other for the configuration server. One is used when a device communicates with the Cisco CNS Configuration Engine using the HTTP protocol. The other one is used when the device communicates with the Cisco CNS Configuration Engine using the publish and subscribe mechanism of the CNS Integration Bus (event bus).

The device must be uniquely identified in these namespaces. The ConfigID uniquely identifies the device in the HTTP domain. The EventID uniquely identifies the device in the CNS event domain.

Because the Cisco CNS Configuration Engine uses both the event bus and the configuration server to provide configurations to devices, both EventID and ConfigID must be defined for each configured Cisco IOS device.

The values for EventID and ConfigID for each device can be identical, or you can make them different when you add or edit device information using the user interface (see "How to Manage Devices" section on page 3-10).

Dynamic ConfigID and EventID Change Synchronization

The Cisco IOS, version 12.2.10T, has been enhanced with new CLI ID commands that can modify the EventID and ConfigID, then reconnect the device to the Cisco CNS Configuration Engine with the new IDs. For example, a device is connected to the Event Gateway with a hostname, say *XYZ*.

The gateway has created a listener for device on events coming on the subscribed subject, say **cisco.cns.config.load.xyz**. There is an entry for this device in the Cisco CNS Configuration Engine directory with attributes, such as IOSEventID and IOSConfigID.

The Cisco CNS Configuration Engine uses IOSEventID to send events to the device. The Cisco CNS Configuration Engine uses IOSConfigID when the device sends an **http get** request to the Cisco CNS Configuration Engine or its configuration template.

If the hostname of the device is changed to, say *ABC*, then the device publishes events **cisco.cns.config.id-changed** and **cisco.cns.event.id-changed** to the Event Gateway. The gateway goes to the directory and fetches the publisher mapping for this subject in the application associated with the group to which the device belongs. The configuration server updates the IOSEventID and IOSConfigID attributes.

Network Management Tools

The CNS 2100 Series platform includes the Tivoli Management Agent (TMA). The Tivoli Product(s) is copyrighted and licensed (not sold) and therefore not transferred.

The owner of the Tivoli Product DISCLAIMS ALL WARRANTIES WITH RESPECT TO THE USE OF THE TIVOLI PRODUCT(S) INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

To initialize the Tivoli Management Agent, see "Initializing Tivoli Management Agent" section on page 2-27.