Diagnostic Tools

Diagnostic Framework

Overview

Unified ICM/Unified CCE servers use the web-based Diagnostic Framework service to collect (and sometimes set) diagnostic information for that server. The Diagnostic Framework service is a REST-like service that accepts requests over HTTPS, gathers information from the system, and responds in the form of an XML response message. It can collect a variety of data, such as process logs, current trace values, network status, PerfMon values, and so on. You can also use the service to collect log files from the server. For a complete list of the capabilities, see Diagnostic Framework API, on page 48.

You can use the Diagnostic Framework as follows:

- For Unified CCE deployments, the primary access method is through the Analysis Manager, which serves as a solution-wide serviceability portal.
- Unified CCE deployments can also use the Unified Communication diagnostic clients' CLI.
- Each Diagnostic Framework service also includes an HTML-based web user interface that provides access to the complete list of the API commands.
- The API can also be accessed directly through a browser.

For more information about how to access the service, see Usage, on page 11.

Installation and Configuration

The Diagnostic Framework service is installed as part of the Unified ICM/Unified CCE software by the ICM-CCE installer (henceforth, called the Unified ICM installer). You require no additional installation or configuration steps. You may optionally choose to customize the service if needed, such as change the port number, certificate, or logging level as explained in the following sections.

Service Registration and Dependencies

Diagnostic Framework is a .NET based web service. It is registered in the Windows service control by the Unified ICM installer.[1] The service files are laid down under the following folder: `<ICM_Drive>:\icm\serviceability\diagnostics`

You can start or stop the Diagnostic Framework service from the Windows service control panel.

The service is registered under the following name:“Cisco ICM Diagnostic Framework”

The Diagnostic Framework is hosted on top of the HTTP service built in the Windows Server kernel. It does not require IIS or any other web server to be installed. The Diagnostic Framework uses the Windows HTTP SSL service to provide secure communications between the server and the client. Therefore, enable the HTTP SSL service before starting the Diagnostic Framework service. The Unified ICM installer configures this dependency in the Windows service control panel to automatically start the HTTP SSL service when you start the Diagnostic Framework service.

Note Note: The Diagnostic Framework or HTTP SSL service does not require IIS. However, if IIS is installed, the HTTP SSL service adds a dependency on the IIS service. Therefore, for HTTP SSL and the Diagnostic Framework to work, start IIS.

Configure Service Port

The Diagnostic Framework listens on TCP port 7890.

You can change the port number. To change the port number, update the Diagnostic Framework service configuration file and the certificate registration with Windows. Change the port number on the CLI and Analysis Manager clients too. Also, change the port number on every other Unified ICM server where other instances of the Diagnostics Framework are running.

Note Consider changing the port number only if necessary.

Procedure

Step 1 Stop Diagnostic Framework service through Windows service control.

Step 2 Open command prompt and change directory to `<ICM_Drive>:\icm\serviceability\diagnostics\bin`

Step 3 Run **DiagFwCertMgr /task:ValidateCertBinding** command and confirm from output that certificate binding with current port is valid.

For more information about the DiagFwCertMgr utility, see Certificate Management, on page 9.

Step 4 Record thumbprint of certificate in use.

[1] The Unified ICM installer detects and installs the appropriate .NET version.

You need the thumbprint to register the certificate with a different port. You can access it either from the output of the preceding command or from the following registry value: `HKLM\SOFTWARE\Cisco Systems, Inc.\ICM\Serviceability\DiagnosticFramework\CertUsedByDiagFwSvc`

Step 5 In same command window, run **DiagFwCertMgr /task:UnbindCert** command to remove certificate binding from current port.

Step 6 Launch Notepad and open service configuration file `<ICM_Drive>:\icm\serviceability\diagnostics\bin\DiagFwSvc.exe.config`

Note You may want to copy this configuration file before you change it.

Step 7 Save file and quit Notepad.

Step 8 Open command prompt and change directory to `<ICM_Drive>:\icm\serviceability\diagnostics\bin`

Step 9 Run **DiagFwCertMgr/task:BindCertFromStore/certhash:<hash of the certificate noted above>** command to bind the certificate to the new port number.

The utility reads the port number from the service configuration file.

Step 10 Read output and confirm that preceding command completed successfully.

Step 11 (Optional) Run **DiagFwCertMgr/task:ValidateCertBinding** command again to verify changes to port number binding.

Step 12 Restart Diagnostic Framework service.

What to do next

If you configured the Windows Firewall, make sure that the new port opened in the firewall configuration.

Enabling ECDSA

Before you begin

Installer generates self-signed ECDSA certificate, imports to the windows local store, and updates the ECDSA thumbprint registry at `SOFTWARE\\WOW6432Node\\Cisco Systems, Inc.\\ICM\\Serviceability\\DiagnosticFramework`.

Procedure

Step 1 In Windows service control, stop Diagnostic Framework service.

Step 2 In the command prompt, change the directory to `<ICM_Drive>:\icm\serviceability\diagnostics\bin`.

You have to remove the binding from the existing port.

Step 3 Run the command **DiagFwCertMgr/task:UnbindCert** to remove the existing certificate binding from the current port. You can now bind the ECDSA certificate.

Step 4 To bind the ECDSA certificate to the current port, run the command **DiagFwCertMgr /task:CreateAndBindCertECDSA**.

Note Certificate matching to the thumbprint of ECDSA registry will be used to bind the port.

To remove ECDSA certificate, you can run the command **DiagFwCertMgr /task:UnbindAndDeleteCertECDSA**. This command will remove the certificate binding from the current port and will delete the self-signed ECDSA certificate created by the option **CreateAndBindCertECDSA**.

Note For more commands of ECDSA, refer, to the table *Diagnostic Framework Certificate Manager Utility Tasks* in the chapter *Diagnostic Tools* at https://www.cisco.com/c/en/us/support/customer-collaboration/unified-contact-center-enterprise/products-installation-and-configuration-guides-list.html

Installing or Updating Third-Party Certificate

During installation, the Diagnostic Framework generates a self-signed certificate with its name set to the server hostname. The self-signed certificate can be replaced with a trusted third-party signed certificate. For more information, see Certificate Management, on page 9.

Diagnostic Framework Log Files and Logging Level

The Diagnostic Framework log files are created in the folder `<ICM_Drive>:\icm\serviceability\diagnostics\logs`.

The Diagnostic Framework uses the industry-standard log4net library to create and manage its log files. A configuration file controls the names of the log files, how large they can get, how many rollover files are kept, the logging level, and so on.

The default logging level 'INFO' is sufficient for most cases. Do not change the logging level unless directed by the TAC.

You can change the log level by editing the file `<ICM_Drive>:\icm\serviceability\diagnostics\config\log4net.config` and changing the <level> tag value to "DEBUG" (or "WARN," "ERROR," or "FATAL").

```
<root>
    <level value="INFO" />
    <appender-ref ref="RollingFileAppender" />
</root>
```

Diagnostic Framework Service Resources Requirements

Reduced Priority

The Diagnostic Framework service runs at a Below Normal priority to avoid adversely impacting server/application performance while running.

Changing Service CPU Threshold

Some CPU-intensive APIs of the Diagnostic Framework first check the overall system CPU utilization value (%CPU). These APIs do not start the request if the %CPU value is greater than a threshold value.

These APIs are:

- LogMgr commands

- TraceMgr commands
- ConfigMgr command

There are a few registry keys that control this behavior. Look in the following Windows Registry Key:

`HKLM\SOFTWARE\Cisco Systems, Inc.\ICM\Serviceability\DiagnosticFramework`

Table 1: CPU Threshold

Registry key	Default Value	Description
CPUThresholdSample	5	To get a more accurate reading of the %CPU, multiple readings are taken. This value says how many samples should be read.
CPUThresholdDelay	2	The number of milliseconds to wait between each sample taken.
CPUThresholdPercent	60	The percent value to compare the current %CPU to. If the %CPU is greater than this value, the API cannot start. An error returns telling the user that the server is too busy, and to try the command later.

Change Maximum Number of Concurrent Requests

The Diagnostic Framework service is designed to handle up to 20 concurrent web requests. The system was tested under load to work with this configuration. However, if you must lower the number of concurrent requests, you can modify the value of maxConcurrentCalls property in the service configuration file.

Procedure

Step 1 Stop Diagnostic Framework service.

Step 2 Launch Notepad and open file `<ICM_Drive>:\icm\serviceability\diagnostics\bin\DiagFwSvc.exe.config`.

Tip You may want to copy this configuration file before you change it.

Step 3 Locate element `<serviceThrottling maxConcurrentCalls="20" />` and change value to any number below 20.

Caution Do not increase the value beyond 20. It may lead to unexpected results during peak call volume.

Step 4 Save file and quit Notepad.

Step 5 Restart Diagnostic Framework service.

Security

The Diagnostic Framework provides the infrastructure to establish a secure connection between the service and its clients. It uses HTTP form authentication over SSL to authenticate, authorize, and encrypt the connection. You need a valid Diagnostic Framework user account to access the service. Connections are not session oriented; the connection is maintained from the receipt of a request until the response is sent.

For service provider deployments, the Diagnostic Framework service is ICM instance aware, and can control access based on instance data requested.

Log In to the Diagnostic Framework Portico

To access the Diagnostic Framework Portico tool, do the following:

Before you begin

If your machine is in a domain, any user who is a local administrator and a domain user on the machine can login.

However, to view the lists and perform the tasks in the Diagnostic Framework Portico tool, you must be a local administrator on the machine *and* must be either a domain admin in the machine's domain or a member of at least one setup security group in the machine's domain.

If your machine is in a workgroup, you must be a local administrator.

Procedure

Step 1 In your browser's address bar, type: **`https://localhost:7890/icm-dp/DignosticPortal`**.

Step 2 Press **Enter**.
The Login page appears.

Step 3 Enter your Active Directory username and password.

Step 4 Click **Log In**.

Log Out of the Diagnostic Framework Portico

For security purpose logout when you are finished using the Diagnostic Framework Portico tool. To log out, click **Log Out** at the top-right of the page. This returns you to the login page.

Note If no activity in which you contact the server occurs in a 30 minute period, you are automatically logged out. If a forced logout occurs, you must log in again to resume using the tool.

Authentication, Authorization, and Auditing

The Diagnostic Framework service integrates with Windows as well as Active Directory to provide user management and access control. The Diagnostic Framework allows two sets of users:

- *A local Windows user who is a member of the local Windows security group called ICMDiagnosticFrameworkUsers on the server where the service exists*: This group is created by the Unified ICM installer and is initially empty, so by default, no local users have access to the service. The administrator on the server can make any local user a member of this group and provide access to Diagnostic Framework service. To add a user to the ICMDiagnosticFrameworkUsers group, use the Computer Management tool under Administrative Tools.

- *A trusted domain user who is a member of local Administrators group on the server where the service exists:* A trusted domain user who is a member of local administrators on the server can make any trusted domain user a member of this group and provide access to Diagnostic Framework service.
- *A trusted domain user who is a member of local ICMDiagnosticFrameworkUsers group on the server where the service exists:* A trusted domain user who is a member of local administrators group on the server can make any trusted domain user a member of ICMDiagnosticFrameworkUsers group and provide access to Diagnostic Framework service.
- *A trusted domain user who is a member of the CONFIG domain security group of the Unified ICM/Unified CCE instance being accessed*: A Unified ICM/Unified CCE SETUP user or domain administrator can make any trusted user a member of the instance CONFIG group. Nested membership is allowed too; as a result the SETUP users and domain administrator can also access the service. To add a user to the instance CONFIG group use the Active Directory Users and Computers tool or Unified ICM/Unified CCE User List tool. Access to domain users is configurable. By default, all direct and nested members of the CONFIG group have access to the service. However, you can disable access to domain users as follows:

  1. Stop the Diagnostic Framework service.
  2. Launch Notepad and open the file `<ICM_Drive>:\icm\serviceability\diagnostics\bin\DiagFwSvc.exe.config`

     Tip Tip: You may want to make a copy of this configuration file before making any changes to it.

  3. Locate the element `<add key="DomainAuthorizationEnabled" value="1" />`and change the value from 1 to 0
  4. Save the file and quit Notepad.
  5. Restart the Diagnostic Framework service.

Note A Diagnostic Framework user does not require administrative privileges on the server to access the service.

The user authentication, validating username and password, is managed by Windows or Active Directory. Therefore, all valid or invalid sign in attempts are logged in the Windows Event Viewer (provided that login/logout auditing is enabled). The user authorization, validating group membership and optionally Unified ICM instance access, is managed by the Diagnostic Framework service. Hence, all authorization requests can be audited through the Diagnostic Framework logs.

Note A user may be a valid Windows or Active Directory user but may not be a member of the required security groups for access to Diagnostic Framework service. As a result, even though the user may pass authentication, it may not pass authorization.

Because the Diagnostic Framework user is managed by Windows or by Active Directory, the user is subjected to the password policies of the server or the domain. Always set strong password policies. For more information about system hardening and password policies, see the *Security Guide for Cisco Unified ICM/Contact Center Enterprise* at https://www.cisco.com/c/en/us/support/customer-collaboration/unified-contact-center-enterprise/products-configuration-examples-list.html.

Special Consideration for Servers with Multiple Unified ICM Instances

This section applies to environments similar to service providers, who have multiple Unified ICM instances on each server.

The domain user is authorized against the CONFIG domain security group of the Unified ICM instance. If there are multiple instances on the server, then the service needs to know which instance security group to authorize against. Therefore, on a multiple Unified ICM instance server, the ICM instance name must be passed as one of the parameters for each request when authorizing a domain user. If an instance name parameter is not passed, then the domain user authorization fails. The local user is free from this requirement because there is only one local group per server. Furthermore, when a domain user is used to access the service, the response is crafted only for the specific instance that user belongs to. However, when a local user tries to access the service, the response includes information for all instances on that server. This gives service providers flexibility to access control information collection for a one or all instances.

On a single instance server, the instance name is not required when you access an API. Because there is only one instance on the server, the domain user is authorized against the CONFIG domain security group of that instance.

The following table summarizes the all authorization combinations. Remember that you can completely disable domain authorization through the service configuration file.

Table 2: Domain Authorization Combination

Unified ICM Instances on Server	User Type	Instance Name Provided	Authorization Criteria	Response Content on Successful Authorization
Multiple	Domain	No	Fail authorization, user must provide instance name in request	HTTP 403 – Access Forbidden
Multiple	Domain	Yes	Authorize against the instance name provided by user	Data for instance requested
Multiple	Local	No	Authorize against local group	Data for all instances
Multiple	Local	Yes	Authorize against local group	Data for instance requested
Single	Domain	No	Automatically detect the instance name and authorize against it	Data for instance installed
Single	Domain	Yes	Authorize against the instance name provided by user. If the instance name is invalid, then authorization fails.	Data for instance installed
Single	Local	No	Authorize against local group	Data for instance installed
Single	Local	Yes	Authorize against local group	Data for instance installed

Encryption

Diagnostic Framework uses SSL to secure the HTTP connection between the server and the client. This secures both the credentials and data exchanged. To establish the SSL connection, the ICM-CCE installer creates a self-signed certificate and uses it during connection negotiation. Because the certificate is self-signed, the browser issues a warning about the invalidity of the certificate trust. Diagnostic Framework allows replacing the self-signed certificate with a trusted third-party certificate. For more information, see the Certificate Management section.

Certificate Management

The ICM-CCE installer creates a self-signed certificate and stores it in the Windows Local Computer Personal certificate store with the friendly name "Cisco ICM Diagnostic Framework service certificate". The installer then binds this certificate to the Windows HTTP service on the Diagnostic Framework service port, which by default is TCP 7890. The Diagnostic Framework service is hosted on top of the Windows HTTP service. Therefore, this certificate is used by Windows HTTP service to establish a secure HTTPS channel (HTTP over SSL) whenever the Diagnostic Framework service is accessed. The Unified ICM installer uses the Diagnostic Framework Certificate Manager Utility to create and bind the self-signed certificate.

Depending on the nature of business and the network access layout of the site, a self-signed certificate may provide sufficient security for accessing the service from within the trusted intranet. However, if you plan to access the service from outside the trusted network, replace the self-signed certificate with a trusted third-party certificate to provide improved security [2] .

When you access the service with the self signed certificate for the first time from a browser, a warning about the validity of the certificate appears. If you are certain that the server is authentic then you may choose to accept the certificate and store it on the client machine to avoid future warnings.

If you wish to replace the server certificate with a trusted third-party certificate or modify the port to which a certificate is bound, you **must** use the Diagnostic Framework Certificate Manager utility.

Diagnostic Framework Certificate Manager Utility

The Diagnostic Framework Certificate Manager utility is a command line utility used to manage certificate creation and binding for the Diagnostic Framework service. It is installed at `<ICM_Drive>:\icm\serviceability\diagnostics\bin\DiagFwCertMgr.exe`.

The utility can perform the following tasks:

- Create self-signed certificate.
- Store the certificate in Local Computer Personal certificate store.
- Bind a certificate to Windows HTTP service on a given port.
- Remove a certificate binding from the Windows HTTP service on a given port.
- Delete the self-signed certificate created by itself from the Local Computer Personal certificate store.
- Validate the certificate binding to HTTP service for Diagnostic Framework service.

The following section explains the usage of the utility:

```
DiagFwCertMgr /task:<task_name> [/port:<port_number>] [/certhash:<certificate_thumbprint>]
 [/logpath:<logfile_path>]
```

[2] A self-signed certificate cannot guarantee the authenticity of the hosting server. Because the client is unaware of the server authenticity, the client should exercise caution when sharing the user credentials with such server. A malicious user may setup a rogue server with a self-signed certificate, claiming to be a legitimate server, and use it to steal user credentials from the client. Always use trusted certificates to authenticate servers when accessing outside your trusted network.

Where:

- `/task:` specifies the task to be performed.
- `/port:` specifies the port number used by the service; this is optional as the port number is automatically read from the service configuration file (DiagFwSvc.exe.config).
- `/certhash:` specifies the SHA-1 thumbprint of the certificate; required only when binding a specific certificate, which exists in the certificate store, to a port.
- `/logpath:` specifies the path where the log file should be created; by default it is the current folder.

The following table explains each task:

Table 3: Diagnostic Framework Certificate Manager Utility Tasks

Task	Description
CreateAndBindCert	Creates a self-signed certificate in the local computer personal certificate store and binds it with HTTP service on the given port. (Used by ICM-CCEInstall)
BindCertFromStore	Looks up the certificate provided by /certhash argument in certificate store and binds it with the HTTP service on the given port.
UnbindCert	Removes the certificate binding from the specified port, does not modify any certificate in the store.
UnbindAndDeleteCert	Removes the certificate binding from the specified port. Also, deletes the self-signed certificate created by CreateAndBindCert option. (Used by ICM-CCE Uninstall)
ValidateCertBinding	Verifies the certificate binding on the specified port and confirms its presence in the local computer certificate store.
CreateAndAddToStoreCertECDSA	Creates and stores the self-signed ECDSA certificate in the local computer certificate store without binding it to the port.
CheckAndCreateStoreCertECDSA	Checks if the certificate is present in the store and creates only if it is NOT present.
CreateAndBindCertECDSA	Creates a self-signed ECDSA certificate in the local computer certificate store and binds it with HTTP service on the given port.
DeleteCertECDSA	Deletes the self-signed ECDSA certificate.
UnbindAndDeleteCertECDSA	Removes the certificate binding from the specified port. Also, deletes the self-signed ECDSA certificate created by **CreateAndBindCertECDS** option.

Diagnostic Framework Certificate Manager utility stores the thumbprint (SHA-1 hash) of the self-signed certificate created by the utility and the certificate used by the Diagnostic Framework service in the registry at the following location respectively:

```
HKLM\SOFTWARE\Cisco Systems, Inc.\ICM\Serviceability\
DiagnosticFramework\SelfSignedCertCreatedForDiagFwSvc
HKLM\SOFTWARE\Cisco Systems, Inc.\ICM\Serviceability\
DiagnosticFramework\CertUsedByDiagFwSvc
```

Unless the certificate used by the service is changed manually, both registry values are the same.

Using a Trusted Third-Party Certificate

Replacing the certificate used by the Diagnostic Framework service involves two tasks. The first task is to import the new certificate in the Local Computer Personal certificate store. The second task is to bind it with the TCP port used by the service.

Import Certificate

Select **`Run`** from the **Start** menu, and then enter **`mmc`**. **MMC** appears.

From the **File** menu, select **`Add/Remove Snap In`**. The **Add or Remove Snap-ins** appears.

Use the MMC Certificates snap-in to import a certificate in the Local Computer Personal certificate store. See Microsoft documention for details on Importing the Certificate into the Local Computer Store.

Caution Diagnostic Framework does not use IIS web server. It is hosted on top of Windows HTTP service. Use the DiagFwCertMgr utility to bind this certificate to the Windows HTTP service.

Bind Certificate

Complete the following instructions to bind the certificate added to the Windows HTTP service using the DiagFwCertMgr utility:

1. Open MMC Certificates snap-in and record the thumbprint of the certificate to use with the Diagnostic Framework service.
2. Stop the Diagnostic Framework service via the Windows service control.
3. Open a command prompt and change directory to `<ICM_Drive>:\icm\serviceability\diagnostics\bin`.
4. In the command window, run the command **DiagFwCertMgr /task:UnbindCert** to remove the current certificate binding from the port.
5. Run the command **DiagFwCertMgr /task:BindCertFromStore /certhash:<hash of the certificate noted above>** to bind the new certificate to the service.

   The utility reads the port number from the service configuration file.
6. Read the output and confirm that the preceding command completed successfully.
7. Optionally, run the DiagFwCertMgr /task:ValidateCertBinding command to verify the changes to the certificate binding.
8. Restart the Diagnostic Framework service.

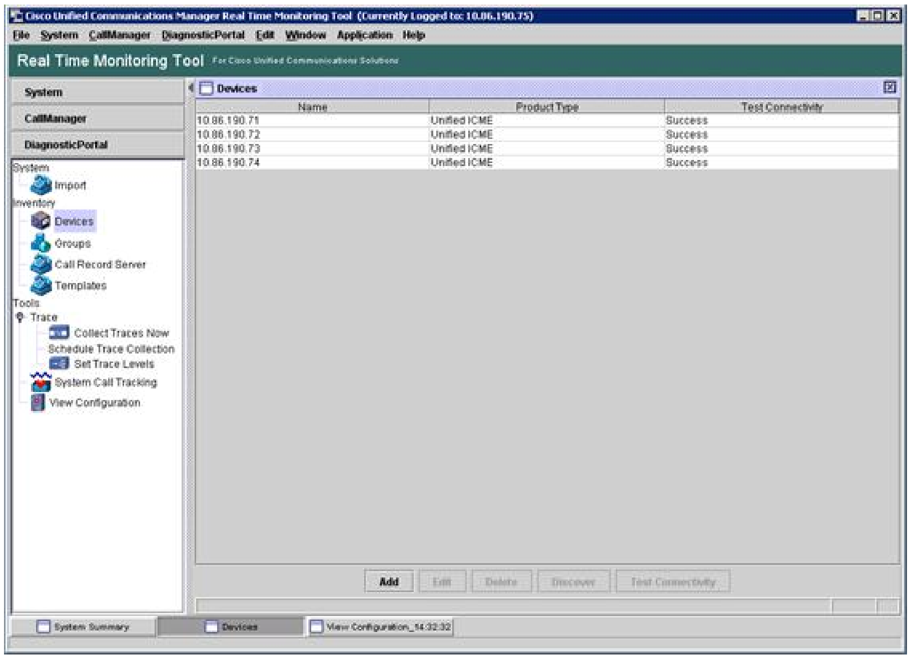
Usage

The framework provides four ways to access the diagnostic data:

Accessing the Diagnostic Framework Through the Analysis Manager

The Analysis Manager is part of the Real Time Monitoring client Tool (RTMT) that resides on Unified CM. RTMT is not a web-based tool, rather it is a thick client tool that you must download from the Unified CM and install on a server. RTMT includes menus for the Analysis Manager.You can access the Analysis Manager functions from the tool. See the sample screen:

Figure 1: Real Time Monitoring Tool


Cisco Unified Communications Manager Real Time Monitoring Tool (Currently Logged to: 10.86.190.75)
File System CallManager DiagnosticPortal Edit Window Application Help
Real Time Monitoring Tool For Cisco Unified Communications Solutions
System
CallManager
DiagnosticPortal
System
Import
Inventory
Devices
Groups
Call Record Server
Templates
Tools
Trace
Collect Traces Now
Schedule Trace Collection
Set Trace Levels
System Call Tracking
View Configuration
Devices
Add
Edit
Delete
Discover
Test Connectivity
System Summary
Devices
View Configuration_14:32:32

Name	Product Type	Test Connectivity
10.86.190.71	Unified ICME	Success
10.86.190.72	Unified ICME	Success
10.86.190.73	Unified ICME	Success
10.86.190.74	Unified ICME	Success

For more information about how to use the Analysis Manager, see the *Cisco Unified Real-Time Monitoring Tool Administration Guide* at https://www.cisco.com/c/en/us/support/unified-communications/unified-communications-manager-callmanager/products-maintenance-guides-list.html.

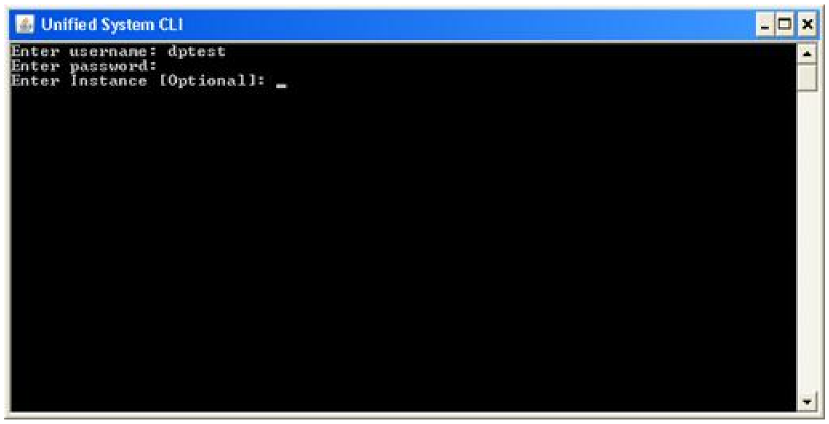
Accessing the Diagnostic Framework Through the Unified System CLI

You can also access the Diagnostic Framework through a CLI. The CLI access utility is installed on every Unified ICM machine at `<ICM_Drive>:\icm\serviceability\wsccli\runwsccli.bat`.

Use a DOS command shell to run this batch file, and it sets up everything needed to access the Diagnostic Framework through the CLI.

A shortcut is included to the Unified ICM menu to provide quick access to the CLI. Also, you can access Unified CLI from **Start** > **Programs** > **Cisco Unified ICM-CCE Tools** > **Unified CLI**. A new DOS Window opens with an initial prompt for your credentials (username and password).

Figure 2: Using Unified System CLI from Command Prompt



```
Enter username: dptest
Enter password:
Enter Instance [Optional]: _
```

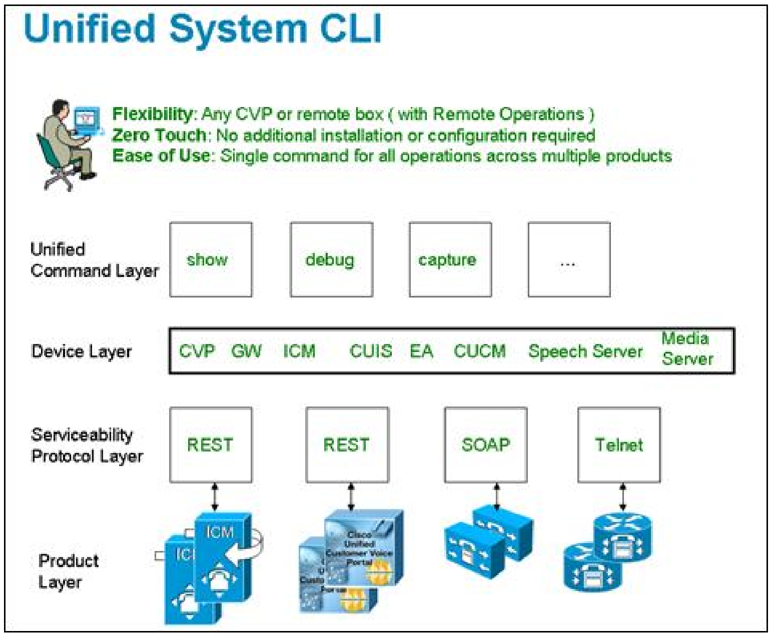
On authentication, you can use the CLI from this window, as explained in Unified CLI Architecture, on page 13.

The CLI allows an optional user input named Instance. In Unified CCE environments, you do not enter anything. In a Hosted environment, you must enter the instance to access the diagnostic data for only that particular instance. Fore more information, see Special Consideration for Servers with Multiple Unified ICM Instances, on page 8.

Unified CLI Architecture

Note This figure is only from a Unified CVP perspective, and does not directly specify the Diagnostic Framework. However, the Diagnostic Framework is what the Unified CCE uses as an underlying implementation.

Figure 3: Unified CLI Architecture


Unified System CLI
Flexibility: Any CVP or remote box (with Remote Operations)
Zero Touch: No additional installation or configuration required
Ease of Use: Single command for all operations across multiple products
Unified Command Layer
show
debug
capture
...
Device Layer
CVP GW ICM CUIS EA CUCM Speech Server Media Server
Serviceability Protocol Layer
REST
REST
SOAP
Telnet
Product Layer
ICM

A user can perform the following tasks using the Unified CLI:

- Run a single command (in system mode) on any Unified CCE system to gather information about all supported solution components.
- In system mode, you can optionally provide the seed devices in WSC_CLI_DIR\conf directory or give a flat CSV file with a device list.
- System mode allows the CLI to recursively go to each supported box in the background and run the same command that the user ran in system mode. User can optionally limit the system command to be run only on certain device group or list of servers. Device group is automatically populated based on device type (Unified CVP, Unified ICM, Cisco IOS Firewall, EA as an example), device IP/hostname wildcard (LOC-1*, 10.86.129.* as an example for branch office deployments), or the CSV file in WSC_CLI_DIR\conf directory.
- You can run the system command by prefixing the “system” on any regular command. For example, “system show all” or typing “system” and running the commands exactly like a regular CLI for interactive mode.

Unified System CLI Usability

- System CLI is automatically installed on all Unified CCE systems as part of the infrastructure, so there is no additional installation required.
- System CLI can be run as a Windows scheduled job or a Unix Cron job. Single command for all operations across multiple products and servers.
- All the commands available in non-system mode for a local system are available in system mode. The command syntax remains the same in system mode. There is an additional option to limit the system command option to certain device group, device type or list of servers.
- In system mode, when you seek help for using the “?” character after you enter the keyword component or subcomponent, the list of components that appears maybe large due to the fact that it is an aggregated list of all the possible component types on all the unique server types.
- The primary list is defined by the unique “Name,” “ProductType.” If there are multiple components for the purpose of co-location, the internal list contains one entry because there is only one WebServices manager running at the specified port.
- System CLI runs on a low priority, so it only uses the IDLE CPU on the System. It should not affect the Call Processing even if it runs on a system working under load. The response time varies depending on the load of the system you are running and the server response time. The response time when there is no running load should be below 5 seconds for each server for simple operations like “version,” “license,” “debug” and “perf.” The response time when there is no running load for “platform” should be below 10 seconds for each server. However, the response time cannot be determined for commands like “trace,” “log,” “sessions,” and all “tech-support” that can vary depending on the data transferred by the server.
- There are no specific timeouts on the System CLI client and it is controlled by the server.
- Error code and error description during failure conditions occur from the server side. System CLI displays the error message arriving from server. The possible error codes are specified and described in the DP REST API specification.

Extensibility

System CLI is not a tool but an extensible platform to build several analysis toolkits. The CLI library can be embedded or used within the analysis engine to do post processing of the data (normalized). System CLI can be used by common scripting tools like Perl to create custom logic.

Command Syntax

The common CLI syntax matches closely with Cisco IOS gateway CLI commands. In cases where specific commands or parameters are not available in IOS gateway, the syntax attempts to match the Unified CM platform CLI commands for consistency.

The following tables list and describe the CLI commands that are available for diagnostic purposes.

Note Ifyou do not specify component/sub-component, then the list includesall the installed components/sub-components on the server.

The command output on screen does not include binary data.

Table 4: CLI Commands

Command (Verb)	Noun	Description
show	all	Aggregation of output for all the supported nounsand specifictothe verb “show.”
	component	Lists the currently installed components on the server.
	configuration	Lists the application configuration.
	debug	Shows the current debug levels.
	license	Shows the license/port information.
	log	Shows the logs.
	perf	Shows the performance information.
	platform	Shows the platform information.
	sessions	Shows the current active sessions/calls. (Not supported by Unified CCE)
	tech-support	Shows system information for Tech-Support. **Note** This command is exactly the same as “show all”.
	trace	Shows the traces.
	version	Shows system hardware and software status and version.
	devices	Shows informationof devices that are known to the CLI.
debug	level	Sets the specific debug level.
help	—	Shows the help information.
quit	—	Quits the CLI.
capture	—	Captures the network packets. (Not supported by Unified CCE)

Note You can enter the start of a command and press **Tab** to complete the command. For example, if you enter **show all comp** and press **Tab**, **show all component** is completed.

You can enter a full command name and press **Tab** to display all the commands or subcommands that are available. For example, if you enter **show** and press **Tab**, you see all the **show** subcommands.

Detailed help that includes a definition of each command and examples of usage is available in the online help.

To get detailed help, at the CLI prompt, enter `help <command>` where *command* specifies the command name or the command and parameter.

To query only command syntax, at the CLI prompt, enter `<command> ?` where *command* represents the command name or the command and parameter.

Note The filter and match features of the CLI are not supported for trace files because the framework returns a zip file that contains not just the text file. For those two features, CLI expects a plain text file.

show all

Syntax

```
show all [options]
```

This command provides information for the component or subcomponent based on the command filters.

Options

component

narrow the output to the specified component(s). The option is limited to trace, debug, perf and sessions commands.

subcomponent

narrow the output to the specified subcomponent(s). The option is limited to trace, debug, perf and sessions commands.

absdatetime

narrow the output to the specified time range in the form of start time and end time. Time format is “mm-dd-yyyy:hh:mm”.

brief

This option is used to prevent the command from collecting certain default logs.

Note This command is used only in Unified CCE to avoid collecting OPC and VRU capture files by default.

reltime

narrow the output to the specified time range in the form of relative time from the current time.

match

narrow the output to the specified regex pattern. This match pattern is applied to text based log output only. The option is limited to trace and log commands.

filter

narrow the output to the specified command(s).

redirect

redirect the output to a file or a directory.

Additional System Mode Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

dtcomponent

narrow the output to the specified component(s) for a device type of the specified component.

dtsubcomponent

narrow the output to the specified subcomponent(s) for a device type of the specified subcomponent.

Examples

show all component cvp:CallServer

show all component cvp:CallServer subcomponent cvp:SIP

show all component cvp:CallServer|cvp:VoiceXMLServer subcomponent cvp:SIP|cvp:VXMLServer

show all component cvp:CallServer subcomponent cvp:SIP filter race|log|version

show all reltime 2 hours

In System Mode

show all devicetype ios

show all devicetype ios|cvp

show all server 10.86.129.11(cvp)

show all group GroupA|default

show all dtcomponent “ucm:Cisco CallManager|cup:Cisco UP SIP Proxy” -- Extract everything from all devices except ucm and cup where device specific filters are applied.

By default, the output zip file is saved at WSC_CLI_DIR\download directory where WSC_CLI_DIR is the environment variable.

To save the output to a specific directory, show all redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show all redirect file `c:\temp\output.txt`

show tech-support

Syntax

```
show tech-support [options]
```

This command provides information for the component or subcomponent based on the command filters similar to the "show all" command.

Options

brief
This option is used to prevent the **show tech-support** command from collecting certain default logs.

Note This command is used only in Unified CCE to avoid collecting OPC and VRU capture files by default.

Example

show tech-support brief

In Unified CCE, this command downloads logs for all components excluding OPC and VRU capture files. In other products, this command behaves the way "show tech-support" command does.

show tech-support brief absdatetime 9-18-2008:14:00 9-20-2008:18:00 redirect C:\temp\

In Unified CCE, this command downloads logs for all components excluding OPC and VRU capture files for the specified start and end time. In other products, this command behaves the way "show tech-support" command does. The output is saved in `c:\temp\clioutput.zip`.

show tech-support component "icm:Peripheral Gateway 1A" subcomponent "icm:opc" absdatetime 9-18-2008:14:00 9-20-2008:18:00 brief redirect C:\temp\

In Unified CCE, this command downloads logs for the component Peripheral Gateway 1A and subcomponent OPC for the specified date and time. The output is saved in `C:\temp\clioutput.zip`. Removing **brief** from the command results in collection of OPC captures as well.

show component

Syntax

```
show component [options]
```

Lists all the installed subcomponents of a component. If component is not given, then all the components and subcomponents configured/installed are listed.

Options

Name of a specific component.

Example

show component cvp:VXMLServer

show config

Syntax

```
show config [options]
```

This command displays the configuration data.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for host names or IP addresses.

group

narrow the output to the specified group name(s).

Example

show config component cvp:CallServer subcomponent cvp:H323

In System Mode

show config devicetype ios

show config devicetype ios|cvp

show config server 10.86.129.11(cvp)

show config group CVPAndIOS|default

To save the output to a directory, show config redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show config redirect file `c:\temp\output.txt`

show debug

Syntax

```
show debug [options]
```

This request returns the current debug level for a component or subcomponent.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for host names or IP addresses.

group

narrow the output to the specified group name(s).

dtcomponent

narrow the output to the specified component(s) for a device type of the specified component.

dtsubcomponent

narrow the output to the specified subcomponent(s) for a device type of the specified subcomponent.

Valid Debug Levels

level 0

Default debug level. During general operation, product log errors or warning trace messages.

level 1

Small performance impact (Warning) debug level. Can be run on production environment. At level 1, additional basic component traces along with level 0 trace messages.

level 2

Medium performance impact (Informational) debug level. Can be run on production environment. At level 2, additional detailed component traces along with level 1 trace messages.

level 3

High performance impact (Debug) debug level. Can be run on production environment. At level 3, most detailed trace messages will be logged along with level 2 trace messages.

level 4

Cannot be run on production environment. At level 4, internal subcomponent trace messages will be logged along with level 3 trace messages.

level 5

Cannot be run on production environment. At level 5, internal functional module trace messages will be logged along with level 4 trace messages.

level 99

Custom debug level. In the case when log levels do not match, 99 will be returned as custom level along data representing the custom debug settings.

Example

show debug component cvp:CallServer

show debug component cvp:CallServer|cvp:VXMLServer subcomponent cvp:H323|cvp:SIP

In System Mode

show debug devicetype cup|ucm|icm

show debug devicetype ios|cvp

show debug server 10.86.129.11(cvp)|10.86.129.123(ucm)

show debug group GroupB|default

show debug dtcomponent “ucm:Cisco CallManager|cup:Cisco UP SIP Proxy|cvp:CallServer”

To save the output to a directory, show debug redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show debug redirect file `c:\temp\output.txt`

show license

Syntax

```
show license [options]
```

This command displays the license data.

Options

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show license

In System Mode

show license devicetype ios|cvp|ucm

show license server 10.86.129.123(ucm)

show license group GroupB|default

To save the output to a directory, show license redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show license redirect file `c:\temp\output.txt`

show log

Syntax

```
show log [options]
```

Displays contents or downloads (if redirect option is used) the product *miscellaneous* log file(s) for a component or subcomponent.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

absdatetime

narrow the output to the specified time range in the form of start time and end time. Time format is “mm-dd-yyyy:hh:mm”.

reltime

narrow the output to the specified time range in the form of relative time from the current time.

match

narrow the output to the specified regex pattern. This match pattern is applied to text based log output only.

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show log component cvp:callserver - displays contents of all the log files for component cvp:callserver; can be a huge output

show log component cvp:vxmlserver absdatetime 9-18-2008:14:00 9-20-2008:18:00 - displays contents of all the log files for component cvp:vxmlserver based on specific start date,time and end date, time values

show log component cvp:vxmlserver absdatetime 9-18-2008:14:00 13:00 - displays contents of all the log files for component cvp:vxmlserver based on specific start date,time and end time values.

show log component cvp:callserver subcomponent sip reltime 10 minutes – displays contents of all the log files based on elapsed time of 10 minutes for component cvp:callserver and subcomponent cvp:sip show log component cvp:callserver absdatetime 9-18-2008:14:00 13:00 match .*CVPS ervlet.* - displays contents of all the log files based on match criteria, time range for component cvp:callserver

show log component cvp:callserver absdatetime 9-18-2008:14:00 13:00 match .*CVPServlet.* redirect file c:\uccelogs - downloads all the log files on match criteria, time range for component cvp:callserver

To save the output to a directory, show log redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show log redirect file `c:\temp\output.txt`

show perf

Syntax

```
show perf [options]
```

This command displays performance data.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

dtcomponent

narrow the output to the specified component(s) for a device type of the specified component.

dtsubcomponent

narrow the output to the specified subcomponent(s) for a device type of the specified subcomponent.

Example

show perf component cvp:CallServer subcomponent cvp:ICM

In System Mode

show perf devicetype ios|cvp

show perf server 10.86.129.11(cvp)

show perf group GroupB|default

To save the output to a directory, show perf redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show perf redirect file `c:\temp\output.txt`

show platform

Syntax

```
show platform [options]
```

Shows information about the operating system and hardware.

Options

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show platform

In System Mode

show platform devicetype ios|cvp|ucm

show platform server 10.86.129.11(cvp)

show platform group GroupB|default

To save the output to a directory, show platform redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show platform redirect file `c:\temp\output.txt`

show sessions

Syntax

```
show sessions [options]
```

This request returns active session status/information.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show sessions component cvp:CallServer subcomponent cvp:IVR

To save the output to a directory, show sessions redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show sessions redirect file `c:\temp\output.txt`

show trace

Syntax

```
show trace [options]
```

Displays contents or downloads (if redirect option is used) the product trace file(s) for a component or subcomponent.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

absdatetime

narrow the output to the specified time range in the form of start time and end time. Time format is "mm-dd-yyyy:hh:mm".

reltime

narrow the output to the specified time range in the form of relative time from the current time.

match

narrow the output to the specified regex pattern. This match pattern is applied to text based log output only.

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

Group

narrow the output to the specified group name(s).

dtcomponent

narrow the output to the specified component(s) for a device type of the specified component.

dtsubcomponent

narrow the output to the specified subcomponent(s) for a device type of the specified subcomponent.

Example

show trace component cvp:callserver - displays contents of all the trace files for component cvp:callserver, can be a huge output

show trace component cvp:vxmlserver absdatetime 9-18-2008:14:00 9-20-2008:18:00 - displays contents of all the trace files for component cvp:vxmlserver based on specific start date,time and end date, time values

show trace component cvp:vxmlserver absdatetime 9-18-2008:14:00 13:00 – displays contents of all the trace files for component cvp:vxmlserver based on specific start date,time and end time values.

show trace component cvp:callserver subcomponent cvp:sip reltime 10 minutes - displays contents of all the trace files based on elapsed time of 10 minutes for component cvp:callserver and subcomponent cvp:sip

show trace component cvp:callserver absdatetime 9-18-2008:14:00 13:00 match .*CVP_7_0_SIP-7.* - displays contents of all the trace files based on match criteria, time range for component cvp:callserver

show trace component cvp:callserver absdatetime 9-18-2008:14:00 13:00 match .*CVP_7_0_SIP-7.* redirect c:\uccelogs - downloads all the trace files on match criteria, time range for component cvp:callserver

To save the output to a directory, show trace redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show trace redirect file `c:\temp\output.txt`

show version

Syntax

```
show version [options]
```

Shows product software version.

Options

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show version

In System Mode

show version devicetype ios|cvp|ucm

show version server 10.86.129.11(cvp)

show version group GroupB|default

To save the output to a directory, show version redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show version redirect file `c:\temp\output.txt`

show devices

Syntax

```
show devices [options]
```

List device information including hostname/ip address and port numbers.

Options

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

Example

show devices

To save the output to a directory, show devices redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, show devices redirect file `c:\temp\output.txt`

debug level

Syntax

```
debug level levelnumber [options]
```

This command is used to set debug level. Valid levels range from integer values between 0 - 5.

Options

component

narrow the output to the specified component(s).

subcomponent

narrow the output to the specified subcomponent(s).

redirect

redirect the output to a file or a directory.

Additional System Options

devicetype

narrow the output to the specified device type(s).

server

narrow the output to the specified device(s).

sysmatch

narrow the output to the list of servers matched with a regexp for hostnames or ip addresses.

group

narrow the output to the specified group name(s).

dtcomponent

narrow the output to the specified component(s) for a device type of the specified component.

dtsubcomponent

narrow the output to the specified subcomponent(s) for a device type of the specified subcomponent.

Debug Levels

level 0

Default debug level. During general operation, product log errors, or warning trace messages.

level 1

Small performance impact (Warning) debug level. Can be run on production environment.At level 1, additional basic component traces along with level 0 trace messages.

level 2

Medium performance impact (Informational) debug level. Can be run on production environment. At level 2, additional detailed component traces along with level 1 trace messages.

level 3

High performance impact (Debug) debug level. Can be run on production environment. At level 3, most detailed trace messages will be logged along with level 2 trace messages.

level 4

Cannot be run on production environment. At level 4, internal subcomponent trace messages will be logged along with level 3 trace messages.

level 5

Cannot be run on production environment. At level 5, internal functional module trace messages will be logged along with level 4 trace messages.

level 99

Custom debug level.In the case when log levels do not match, 99 will be returned as custom level along data representing the custom debug settings.

Example

debug level 1 component cvp:CallServer

debug level 2

debug level 99 custom app-defined-data component cvp:callserver subcomponent cvp:sip

In System Mode

debug level 0 devicetype cup|ucm|icm

debug level 1 devicetype ios|cvp

debug level 2 server 10.86.129.11(cvp)|10.86.129.123(ucm)

debug level 3 group GroupB|default

debug level 3 dtcomponent "ucm:Cisco CallManager|cup:Cisco UP SIP Proxy|cvp:CallServer"

To save the output to a directory, debug level 1 redirect dir `c:\temp\` -- the output is saved in `c:\temp\clioutput.zip`.

To save the output to a text file, debug level 1 redirect file `c:\temp\output.txt`

System Mode Syntax

Following is the system mode syntax.

Note You can add product specific extensions; however. any extension must be reviewed by this common cross-product team for clarity and consistency.

Table 5: System Mode Syntax

Command (Verb)	Noun	Description
system		Enter the interactive system mode of the CLI. Use quit/exit command to exit the system mode.

System Command (show all)

Syntax

```
show all
```

The system command can also be run by prefixing the “system” on any regular command for non-interactive mode. For example, “system show all”.

Parameters

[component *component(s)*] [subcomponent *subcomponent(s)*] [filter *noun(s)*] [absdatetime *startdatetime enddatetime*] [reltime <value> minutes/hours/days/weeks/months] [match <string value>] [| <output modifier>] **[group *group(s)*] [server *server(s)*][sysmatch <string value>][devicetype <product type>]**

Note The options highlighted in bold above are included to commands in system mode.

Options

group

narrows the output to selected group(s) only.

server

narrows the output to selected server(s) only.

sysmatch

match a particular string as specified by <string value>.

Note The command notifies about a possible impact to system performance and asks you if you want to continue.

Warning Because running this command can affect system performance, run the command during off-peak hours.

Aggregation of output for all the supported nouns and specific to the verb “show”.

Example-1

```
admin:system
admin(system):show all redirect dir c:\system-tech-support
[server-1]
   server-1 show all Output
[server-2]
   server-2 show all Output
[server-3]
   server-3 show all Output
[server-4]
   server-4 show all Output
[server-5]
   server-5 show all Output
[server-6]
   server-6 show all Output

Output is saved to "c:\system-tech-support\clioutput0.zip"
```

Example-2

Assuming Group:Branch-1 contains server-2, server-3 and Group:Branch-2 contains server-5, server-6

```
admin:system
admin(system): show all group Branch1 | Branch2 redirect dir c:\system-tech-support
[server-2]
   server-2 show all Output
[server-3]
   server-3 show all Output
[server-5]
   server-5 show all Output
[server-6]
   server-6 show all Output

Output is saved to "c:\system-tech-support\clioutput0.zip"
```

Example-3

```
admin:system
admin(system):show all server server-1 | server-6 redirect dir c:\system-tech-support
[server-1]
   server-1 show all Output
[server-6]
   server-6 show all Output

Output is saved to "c:\system-tech-support\clioutput0.zip"
```

Example-4

Assuming that server-2, server-3, server-5 are in subnet 10.86.129.xxx

```
admin:system
admin(system):show all group Branch1 | Branch2 sysmatch redirect dir c:\system-tech-support

[server-2]
   server-2 show all Output
[server-3]
   server-3 show all Output
```

```
[server-5]
   server-5 show all Output

Output is saved to "c:\system-tech-support\clioutput0.zip"
```

Example-5

```
admin:system show all redirect ftp://vpalawat:password/SR609140000
[server-1]
   server-1 show all Output
[server-2]
   server-2 show all Output
[server-3]
   server-3 show all Output
[server-4]
   server-4 show all Output
[server-5]
   server-5 show all Output
[server-6]
   server-6 show all Output

Output is saved to "ftp-sj.cisco.com\incoming\SR609140000-0.zip"
Output is saved to "ftp-sj.cisco.com\incoming\SR609140000-1.zip"
```

Example-6

Assuming that devices configured in OAMP are CVP[server-5], IOS [server-2, server-3], UCM [server-4] and ICM [server-1] .

```
admin:system
admin(system):show all devicetype cvp|ios redirect dir c:\system-tech-support
[server-2]
   server-2 show all Output of ProductType [ios]
[server-3]
   server-3 show all Output of ProductType [ios]
[server-5]
   server-5 show all Output of ProductType [cvp]

Output is saved to "c:\system-tech-support\clioutput0.zip"
```

Run Automated Commands

CLI or System CLI commands can be run automatically using the following mechanism:

- Create a batch file with the commands given below as an example:

  ```
  REM VERSION-COLLECTION
  echo system show version redirect dir c:\test\ > clicmds.txt
  echo exit >> clicmds.txt
  type clicmds.txt | wsccli.bat inplace nointeractive "user:wsmadmin" "passwd:<password>"
  ```

- To define a multiple component and sub-component filter, use double quotes as follows:

  ```
  REM CONFIG-COLLECTION
  echo show config comp CallServer subc "SIP|ICM" redirect dir c:\test\ > clicmds.txt
  echo exit >> clicmds.txt
  type clicmds.txt | wsccli.bat inplace nointeractive "user:wsmadmin" "passwd:<password>"
  ```

- Automated trace collection on CVP servers using a scheduled job:

```
REM TRACE-COLLECTION
echo show trace device cvp redirect dir c:\test\ > clicmds.txt
echo exit >> clicmds.txt
type clicmds.txt | wsccli.bat inplace nointeractive "user:wsmadmin" "passwd:<password>"
```

- Automated script can be invoked from a Windows scheduled job for automated tasks.

Note Note: Because running the automated commands and non-interactive mode can affect system performance, run the command during off-peak hours.

Import File Syntax

The file to be imported is `<ICM_Drive>:\icm\serviceability\wsccli\conf\devices.csv`.

A sample file named devices-sample.csv is provided. Add the devices to this file, and then restart the Unified System CLI to load those devices.

Devices CSV File Syntax

```
##################################################################################
# Sample CSV file for importing devices. File name should be devices.csv
# The file should be in the WSC_CLI_DIR/conf folder
#
# The possible values for Product Type are given below:
#
# * UCM - For Unified CM
# * CVP - For Unified CVP
# * ICM - For Unified ICME, Unified ICM
# * UCCX- For Unified CCX
# * IOS  - For IOS Gateway
# * EA  - For Unified Expert Advisor
# * CUIC  - For Unified Intellegence Center
# * CUP  - For Unified Presence ( that includes the SIP Proxy )
##################################################################################
#
# The column assignments are as follows:
#
# HOSTNAME    -- Mandatory
# DESCRIPTION
# PRODUCT_TYPE   -- Mandatory
# GROUP
# USERNAME
# PASSWORD
# PORT_NUMBER   -- Mandatory
# ENABLE_PASSWORD
# IS_SEED_SERVER
#
HOSTNAME, DESCRIPTION, PRODUCT_TYPE, GROUP, USERNAME, PASSWORD, PORT_NUMBER,
ENABLE_PASSWORD,
IS_SEED_SERVER #10.86.129.109, IOS GW, IOS, Location_1, cisco, cisco, 22, cisco,
```

Note All references to ICM in the above text file equal Unified CCE.

Device, Protocol and Command Mapping Table

The mapping table for device type, command, and serviceability protocol created in WSC_CLI_DIR/conf folder is as follows:

Table 6: Device, Protocol, and Command Mapping

	CVP	Unified CCE	EA	CUIC / LiveData	Speech Server	Media Server	Trace Server	IOS GW	Unified CM	Unified CCX	Finesse
capture	REST	✖	✖	✖	REST	REST	REST	✖	✖	✖	✖
config	REST	REST	REST	✖	✖	✖	✖	TELNET/ SSH	✖	✖	✖
debug	REST	REST	REST	✖	✖	✖	✖	TELNET/ SSH	SOAP	REST	✖
license	REST	REST	REST	✖	❓	✖	✖	TELNET/ SSH	SOAP	REST	✖
log	REST	REST	✖	✖	REST	REST	REST	✖	✖	✖	✖
perf	REST	REST	✖	✖	REST	REST	REST	TELNET/ SSH	✖	✖	✖
platform	REST	REST	SOAP	SOAP	REST	REST	REST	TELNET/ SSH	SOAP	SOAP	SOAP
sessions	REST	✖	✖	✖	✖	✖	✖	TELNET/ SSH	✖	✖	✖
trace	REST	REST	SOAP REST	SOAP REST	❓	❓	❓	TELNET/ SSH	SOAP	SOAP REST	SOAP
version	REST	REST	SOAP REST	SOAP REST	REST	REST	REST	TELNET/ SSH	SOAP	SOAP REST	SOAP

✖— Not supported ❓— Actual

Note Cisco Finesse does not support System CLI for system trace settings.

Note By default from the release 12.5(2) onwards, for IOS GW in the sample devices CSV file the port number is 22. When you want to use Telnet modify the port number to 23.

CLI has the primary list of all devices from seed servers. It runs the system command on each device recursively based on the protocol supported in this release and according to the mapping table given above.

Primary list is defined by the unique “Name”, “ProductType”. If there are multiple devices for the purpose of co-location, the internal list still contains one entry for a product type because there is only one WebServices manager running at the specified port.

CLI also pulls the component/sub-component list from all the devices to create a primary list dynamically.

The CLI output is in the structure of **`[Server]/[Type]/clioutput`**. A single (or multiple zip in case exceeding the size of zip file of 1GB) zip file is created for the aggregate response from all servers.

Mapping of System CLI Commands to IOS CLI Commands

Note This mapping table is available in the configuration file, so that mapping can be easily altered.

Table 7: Mapping of System CLI Commands to IOS CLI Commands

System CLI	IOS CLI
“show config”	“show running-config”
“show version”	“show version”
“show license”	“show license”
“show perf”	“show call resource voice stat” “show memory statistics” “show processes cpu history” “show processes memory sorted” “show voice dsp group all” “show voice dsp voice”
“show debug”	“show debug”
“show log”	N/A
“show sessions”	“show call active voice compact”
“show tech-support”	“show tech-support” <Everything else given above>
“show trace”	“show logging”
“show platform”	“show diag”

“debug”	`0 no debug all`
	`1 -`
	`deb ccsip err`
	`deb cch323 err`
	`deb voip app vxml err`
	`deb http client err`
	`deb mrcp err`
	`deb rtsp err`
	`deb h225 asn1 err`
	`deb h245 asn1 err`
	`2 -`
	`debug isdn q931`
	`debug h225 events`
	`debug h245 events`
	`debug voip ccapi inout`
	`debug vtsp events`
	`3 -`
	`debug ccsip messages`
	`debug h225 q931`
	`debug h225 asn1`
	`debug h245 asn1`

Logs

You can find all logs generated by the CLI process under the directory `<ICM_Drive>:\icm\serviceability\wsccli`.

Accessing the Diagnostic Framework Through the Built-In User Interface (Portico)

Note Starting release 12.6(2), Diagnostic Framework will use only HTTPS to communicate with Framework.

For an end-user to easily harness the functionality of the Diagnostic Framework, a built-in, web-based menu utility called the Diagnostic Framework Portico, allows a user to interact with the framework through their browser. On successful login with the url (**https://localhost:7890/icm-dp/DiagnosticPortal**) it generates an HTML page that can be used to interactively create framework requests and view their replies from the Diagnostic Framework in the same page for the specified server.

Users who do not have access to the Analysis Manager can use this command to gather data from the Diagnostic Framework, without having to know all of the API URLs and parameter values. The Diagnostic Portico Home page recognizes and supports machines with multiple instances [Hosted environment] installed. To access the Diagnostic Framework homepage, no special client side files or installations are needed. You can access the Diagnostic Framework Portico Login page from any machine with a compatible browser.

The entry point for Diagnostic Portico home and menu is through login page. The URL to access is as follows: `https://localhost:7890/icm-dp/DiagnosticPortal`

Where <*UCCE-server*> is the hostname or IP address of the desired server, and <*port*> is the access port (usually 7890).

You can also access the Diagnostic Framework Portico by choosing All Programs > Cisco Unified CCE Tools > Diagnostic Framework Portico.

Most of the commands return simple XML data; the menu utility does some XML parsing and displays the results. A few of these commands create links to allow the user to download the returned files.

The Portico dynamically updates and displays recent changes to processes as below:

- for a process restarted in the last ten minutes, the uptime is underlined and highlighted in red.
- for a process restarted more than 10 minutes ago but less than 30 minutes ago, the uptime is yellow.
- when the status of a process as defined inside the parentheses changes, the process is bolded and highlighted in blue for 10 minutes or until it returns to its former state.

Accessing Diagnostic Framework Commands Through a Browser

Because the Diagnostic Framework is a XML/HTTP based REST-style RPC referred as “RPC-Hybrid” interface, you can access the Diagnostic Framework commands directly through a browser. To access the commands from a browser, type the full URL of the desired command, at the browser address location.

For example, the following URL:

```
https://<UCCE-Server>:<port>/icm-dp/rest/DiagnosticPortal/GetTraceLevel?Component=
Component/Subcomponent
```

The browser displays the data in XML or may ask you to save the file if you are downloading the file. For more information about the URL, see Diagnostic Framework API, on page 48.

The complication with this technique is that there are many APIs, and many of them contain various parameters that you must properly specify.

CLI Configuration

This section will walk you through the configuration required to enter “System mode” and access all devices in your deployment from a single system CLI console window. The CLI supports the following devices:

- All UCCE servers (Routers, Loggers, PGs, ADS, and so on)
- CVP
- CUPS
- Gateways
- UCM
- IP IVR
- CUIC
- Finesse

There are two methods to configuring System mode in the CLI. The method used will depend on whether or not the environment contains CVP OAMP. Customers without CVP OAMP can still utilize the CLI using a CSV file for connection information.

Deployment Option 1: CVP OAMP

CVP OAMP deployment options has several advantages over using Devices.csv including

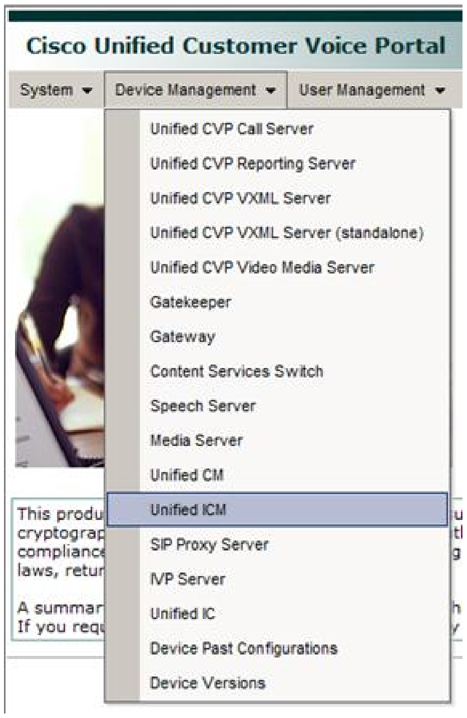
- All devices are centrally added to and stored in CVP OAMP. One update on OAMP will be reflected in all CLI clients.
- Passwords for devices are encrypted in OAMP.
- CVP Remote Operations can be installed on any Windows machine, such as a personal laptop, simplifying setup and access to all devices.

Configure System CLI with CVP OAMP

The first step for setting up System mode is to add all of the devices in your deployment to CVP OAMP.

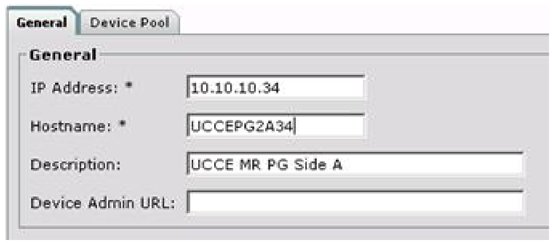
Procedure

Step 1 Sign in to CVP Operations Console from a web browser and select **Device Management** > **Unified ICM**.



Step 2 Click **Add New**.

Step 3 Enter settings for **IP Address**, **Hostname**, and **Description** fields.



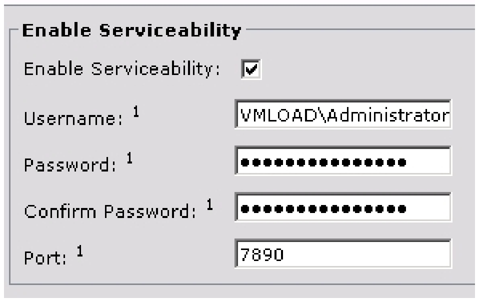
General

IP Address: *	10.10.10.34
Hostname: *	UCCEPG2A34
Description:	UCCE MR PG Side A
Device Admin URL:	

Step 4 Check **Enable Serviceability**.

Step 5 Enter **Username** and **Password** fields with sign-in credentials for that particular device.

Leave the default port as 7890.

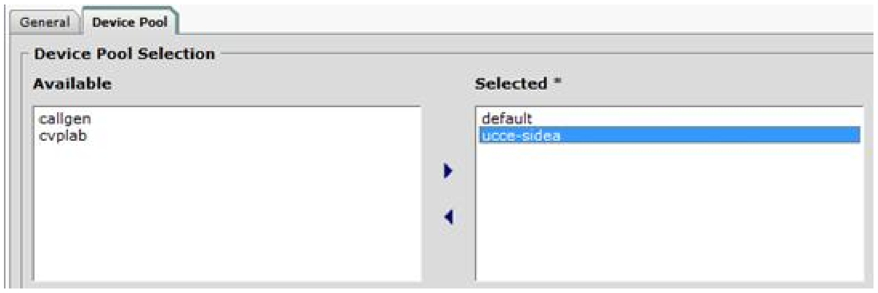


Enable Serviceability

Enable Serviceability:	✓
Username: 1	VMLOAD\Administrator
Password: 1	•••••••••••••••
Confirm Password: 1	•••••••••••••••
Port: 1	7890

Step 6 (Optional) Click **Device Pool** tab and associate the device.

Tip Create a “UCCE-SideA” group for all devices on the A-side.



Step 7 Click **Save**.

What to do next

Repeat the above process for all other devices such as UCCE, CUIC, UCM, Gateways, etc.

Modify or Add User to CVP OAMP for System CLI

By default on installation, the user "wsmadmin" is created with the same password as the OAMP Administrator user. If you wish to modify the password for this user, or create a new user, follow these steps:

Procedure

Step 1 Click **User Management** > **Users** in CVP Operations Console.

Step 2 Add or modify user.

- To modify user, click **wsmadmin** in the List of Users.
- To add user, click **Add New**.

Step 3 Once new username and/or password has been entered, click **User Groups** tab and add "ServiceabilityAdministrationUserGroup" to "Selected" bucket on right side.

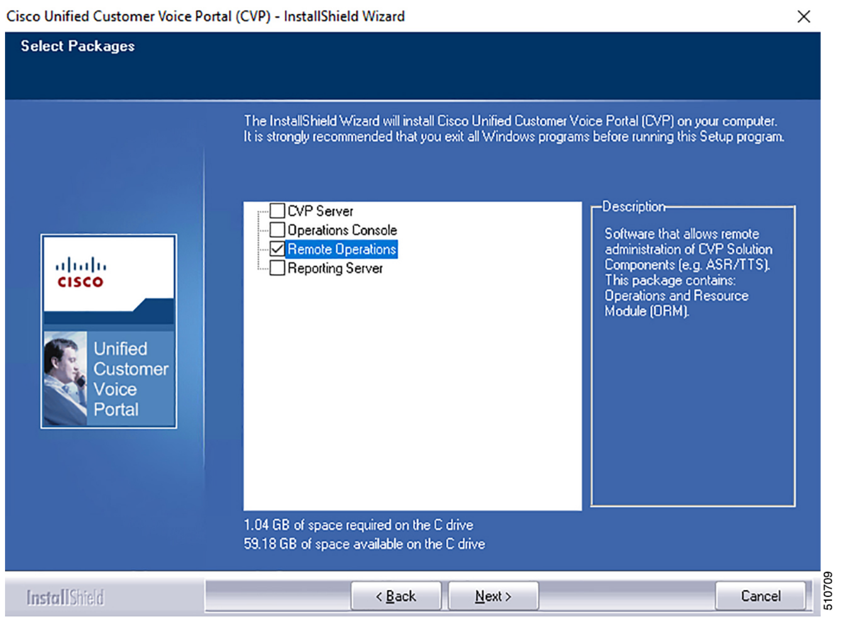
Step 4 Click **Save**.

Install CVP Remote Operations

Once all devices are added to OAMP, you then need to install the CLI on the system from which you intend to access them. The CVP Installer's "Remote Operations" package automatically includes the System CLI.

Procedure

Step 1 Run CVP Installer and select **Remote Operations** checkbox.


Cisco Unified Customer Voice Portal (CVP) - InstallShield Wizard

Select Packages

The InstallShield Wizard will install Cisco Unified Customer Voice Portal (CVP) on your computer. It is strongly recommended that you exit all Windows programs before running this Setup program.

- [ ] CVP Server
- [ ] Operations Console
- [x] Remote Operations
- [ ] Reporting Server

Description

Software that allows remote administration of CVP Solution Components (e.g. ASR/TTS). This package contains: Operations and Resource Module (ORM).

1.04 GB of space required on the C drive
59.18 GB of space available on the C drive

Step 2 Apply security hardening if desired and complete installation.

Add Remote Operations Machines to CVP Operations Console

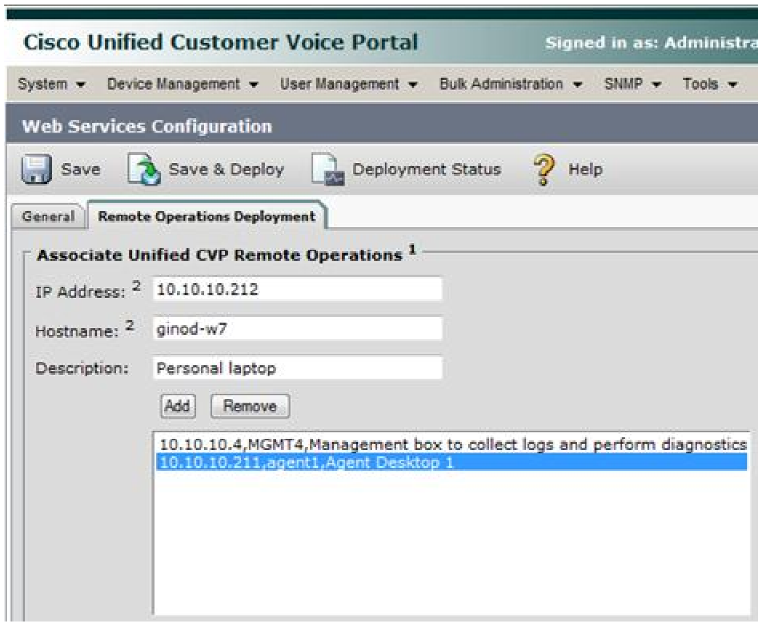
Procedure

Step 1 Sign into CVP Operations Console.

Step 2 Select **System** > **Web Services**.

Step 3 Click **Remote Operations Deployment** tab.

Step 4 Enter remote operations deployment settings for all remote operations machines.



Associate Unified CVP Remote Operations 1

IP Address: 2 10.10.10.212

Hostname: 2 ginod-w7

Description: Personal laptop

10.10.10.4,MGMT4,Management box to collect logs and perform diagnostics
10.10.10.211,agent1,Agent Desktop 1

a) Enter IP address and host name of machine where CVP Remote Operations is installed.
b) (Optional) Enter description.
c) Click **Add**.

Step 5 Click **Save & Deploy** to make devices available for Remote Operations.

You will be informed that the Web Services configuration deployment is in progress.

Step 6 Click **Deployment Status** button to verify status of newly-added machine(s).

Step 7 Click **Refresh** button until status changes to “Success”.

Confirm Windows Environment Variables Set Correctly for CVP Web Services

This should have been taken care of by the CVP Remote Operations installation but intermittently fails, so it is important to verify before attempting to connect to the CLI.

Procedure

Step 1 Click **Start** > **Run** and enter **systempropertiesadvanced** on the Remote Operations machine.

Step 2 Click **Environment Variables**.

Step 3 Verify system variable *WSC_CLI_DIR* is set to `C:\Cisco\CVP\wsm\CLI`.

Step 4 Verify path variable contains `C:\Cisco\CVP\wsm\CLI;`.

Use Unified System CLI with CVP OAMP

Now that the configuration is finished, you are ready to sign in to the CLI and enter System mode.

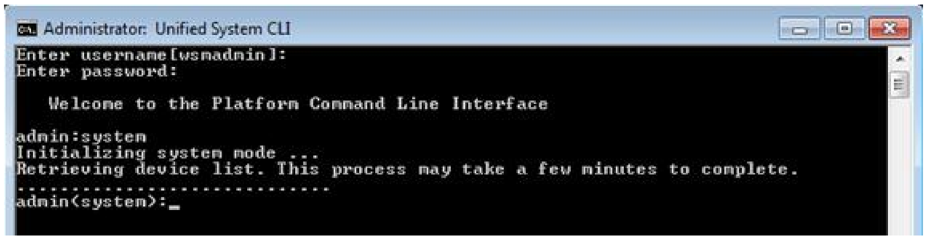
Procedure

Step 1 Select **Start** > **Programs** > **Cisco Unified Customer Voice Portal** > **Unified System CLI** to open Unified System CLI on Remote Operations machine.

Step 2 Sign in with user "wsmadmin" (or sign in with the new user).

Step 3 Type **system** to enter System mode.

Servers that are successfully discovered are indicated by a "."; servers not discovered are indicated by "Unable to connect". Once initial connection is complete, (system) will be displayed in the command prompt. All commands entered while in System mode will be run against all reachable devices defined in CVP OAMP.



```
Enter username[wsmadmin]:
Enter password:

   Welcome to the Platform Command Line Interface

admin:system
Initializing system mode ...
Retrieving device list. This process may take a few minutes to complete.
...............................
admin(system):_
```

What to do next

Any changes made in OAMP while a CLI session is active will not be reflected immediately. There are two options for receiving the updates:

- Close console window and start new connection.
- Type "exit" to leave System mode and then "system init".

Deployment Option 2: Devices.csv

When CVP is not present, Unified System CLI requires a devices.csv file to be configured on the local machine in order to enter System mode. This file contains connection information for all devices in the deployment that should be reachable by the single CLI window.

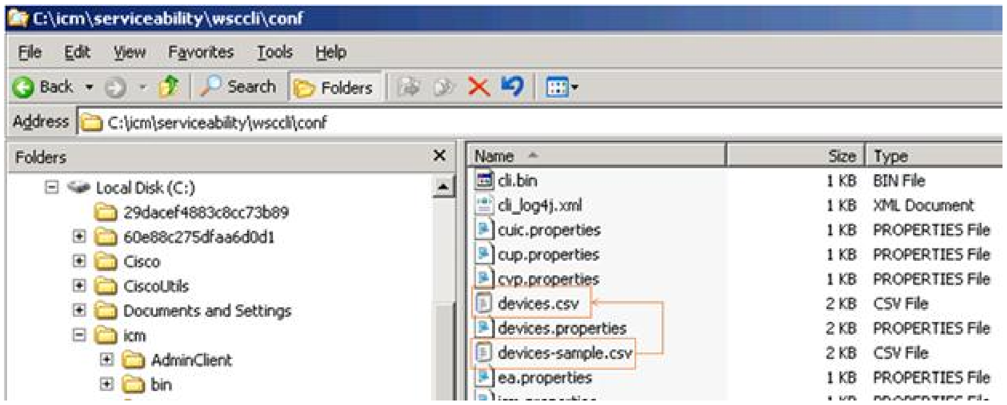
We will use the ADS as our main machine for running the System CLI.

Create Devices.csv from Sample File

Procedure

Step 1 Navigate to `C:\icm\serviceability\wsccli\conf\`.

Step 2 Copy file `devices-sample.csv` and save as `devices.csv`.



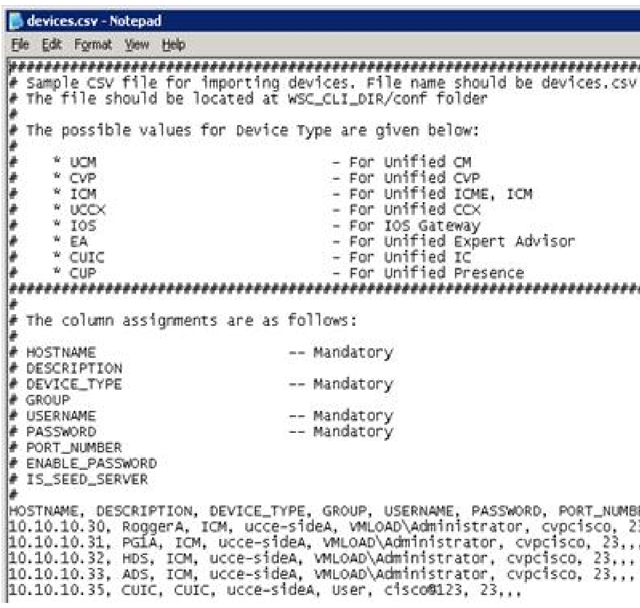
Name	Size	Type
cli.bin	1 KB	BIN File
cli_log4j.xml	1 KB	XML Document
cuic.properties	1 KB	PROPERTIES File
cup.properties	1 KB	PROPERTIES File
cvp.properties	1 KB	PROPERTIES File
devices.csv	2 KB	CSV File
devices.properties	2 KB	PROPERTIES File
devices-sample.csv	2 KB	CSV File
ea.properties	1 KB	PROPERTIES File

Add Connection Information to Devices.csv File

Each device must be added on its own line at the bottom of the `devices.csv` file.

Procedure

	Command or Action	Purpose
Step 1	Within each line you must specify the following required fields: • IP address and hostname • Device Type (from the options listed at the top of the file) • Username • Password • Port Number (leave the default 23 in most cases)	
Step 2	In addition, specifying the following fields make usage easier: • Description • Group (for example, UCCE-SideA)	

	Command or Action	Purpose
Step 3	Save `devices.csv` when complete.	

```
devices.csv - Notepad
File  Edit  Format  View  Help
##############################################################################
# Sample CSV file for importing devices. File name should be devices.csv
# The file should be located at WSC_CLI_DIR/conf folder
#
# The possible values for Device Type are given below:
#
#     * UCM                          - For Unified CM
#     * CVP                          - For Unified CVP
#     * ICM                          - For Unified ICME, ICM
#     * UCCX                         - For Unified CCX
#     * IOS                          - For IOS Gateway
#     * EA                           - For Unified Expert Advisor
#     * CUIC                         - For Unified IC
#     * CUP                          - For Unified Presence
##############################################################################
#
# The column assignments are as follows:
#
# HOSTNAME                     -- Mandatory
# DESCRIPTION
# DEVICE_TYPE                  -- Mandatory
# GROUP
# USERNAME                     -- Mandatory
# PASSWORD                     -- Mandatory
# PORT_NUMBER
# ENABLE_PASSWORD
# IS_SEED_SERVER
#
HOSTNAME, DESCRIPTION, DEVICE_TYPE, GROUP, USERNAME, PASSWORD, PORT_NUMB
10.10.10.30, RoggerA, ICM, ucce-sideA, VMLOAD\Administrator, cvpcisco, 2
10.10.10.31, PG1A, ICM, ucce-sideA, VMLOAD\Administrator, cvpcisco, 23,,
10.10.10.32, HDS, ICM, ucce-sideA, VMLOAD\Administrator, cvpcisco, 23,,,
10.10.10.33, ADS, ICM, ucce-sideA, VMLOAD\Administrator, cvpcisco, 23,,,
10.10.10.35, CUIC, CUIC, ucce-sideA, User, cisco@123, 23,,,
```

Designate Users for Diagnostic Framework

Users must be a part of the Local Group “ICMDiagnosticFrameworkUsers” in order to initially sign in to the CLI when using devices.csv.

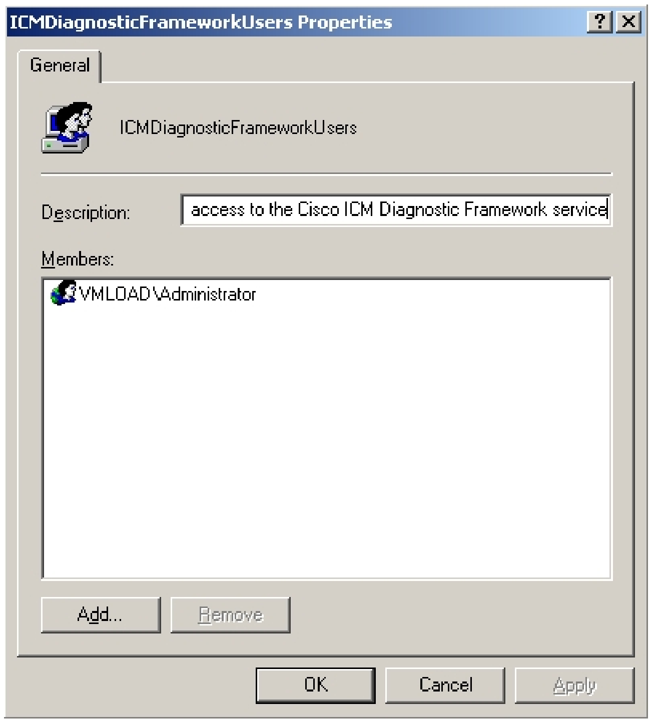
Procedure

Step 1 Click **Start** > **Run**.

Step 2 Enter “lusrmgr.msc”

Step 3 Open **Groups** folder and double-click **ICMDiagnosticFrameworkUsers**.

Step 4 Add users to group and click **OK**.


ICMDiagnosticFrameworkUsers Properties
General
ICMDiagnosticFrameworkUsers
Description:
access to the Cisco ICM Diagnostic Framework service
Members:
VMLOAD\Administrator
Add...
Remove
OK
Cancel
Apply

Use Unified System CLI with Devices.csv

Procedure

Step 1 Select **Start** > **Programs** > **Cisco Unified CCE Tools** > **Unified System CLI** on ADS.

If this shortcut is missing for some reason, run **C:\icm\serviceability\wsccli\runwsccli.bat**.

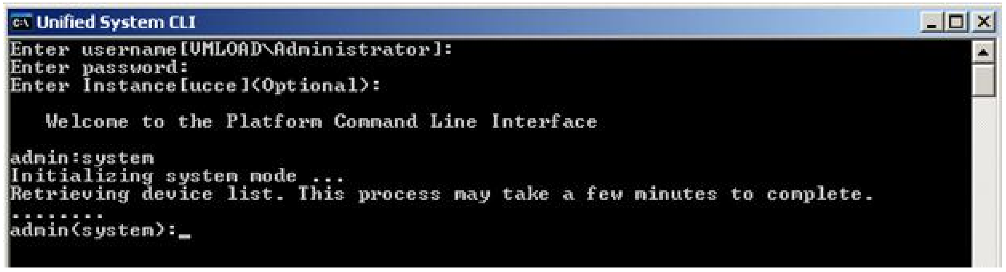
Step 2 Sign in with member of ICMDiagnosticFrameworkUsers group.

If you receive an immediate “Unable to connect to localhost:7890(icm)” error, the Diagnostic Framework service may not be running. Click **Start** > **Run** and enter **services.msc**. Ensure “Cisco ICM Diagnostic Framework” is started.

Step 3 Once successfully signed in to local machine, type **system** to enter System mode.

Servers successfully discovered are indicated by a “.” and those that cannot be reached are indicated by “Unable to connect”.

Once initial connection is complete, “(system)” will be displayed in the command prompt. All commands entered while in System mode will be run against all reachable devices defined in devices.csv



```
Unified System CLI
Enter username[VMLOAD\Administrator]:
Enter password:
Enter Instance[ucce](Optional):

   Welcome to the Platform Command Line Interface

admin:system
Initializing system mode ...
Retrieving device list. This process may take a few minutes to complete.
........
admin(system):_
```

Running the System CLI from Multiple Machines with Devices.csv

If you intend to run the System CLI on another machine, such as a second ADS, the `devices.csv` file must be copied to that second machine. Any changes made to one `devices.csv` will need to be manually made on the additional machines as well.

Diagnostic Framework API

The Diagnostic Interface supports the following commands.

GetTraceLevel

The Diagnostic Framework supportsfour levels of trace configuration based on level of trace detail and performance impact; the Diagnostic Framework translates the following levels to component- or process-specific trace level settings:

Table 8: Trace Levels

Trace Level	Description
0	Product/component install default, should have no/minimal performance impact
1	Less detailed trace messages, small performance impact
2	More detailed trace messages,medium performance impact
3	If the trace level does not match any pre-defined levels (for example, a manually configured, specific trace mask), Diagnostic Framework returns “custom (99)”.

Note The minimum and default trace level for the CMS, CMSJServer and ISE components is 2.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetTraceLevel?Component=
Component/Subcomponent
```

Reply example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dp:GetTraceLevelReply ReturnCode="0"
 xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0"/>
<dp:Trace Level="0"/>
</dp:GetTraceLevelReply>
```

SetTraceLevel

For more information about the trace level values, see GetTraceLevel, on page 48.

Request:

```
 https://<server>:<port>/icm-dp/rest/DiagnosticPortal/SetTraceLevel?Component=
Component/Subcomponent&Level=1
```

Reply example:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<dp:SetTraceLevelReply ReturnCode="0"xmlns:dp="http://www.cisco.com/vtg/DiagnosticPortal">
```

```
<dp:Schema Version="1.0"/>
```

```
</dp:SetTraceLevelReply>
```

ListTraceComponents

Lists all possible application components that produce trace files. Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/ListTraceComponents
```

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:ListTraceComponentsReply ReturnCode="0"
xmlns:dp="http://www.cisco.com/vtg/DiagnosticPortal">
<dp:Schema Version="1.0" />
<dp:TraceComponentList>
<dp:TraceComponent Name="Logger A" ComponentType="Logger" Description="ICM Component"
IsLevelConfigurable="true"
  IsFileCollectable="true">
 <dp:TraceComponentList>
 <dp:TraceComponent Name="baImport" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="CampaignManager" Description="ICM Process for Component LoggerA"

  IsLevelConfigurable="true" IsFileCollectable="true" />
 <dp:TraceComponent Name="clgr" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="csfs" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="cw2kFeed" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="dtp" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="hlgr" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="nm" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="nmm" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="rcv" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="rpl" Description="ICM Process for Component LoggerA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
</dp:TraceComponentList>
</dp:TraceComponent>
<dp:TraceComponent Name="Router A" ComponentType="Router" Description="ICM Component"
IsLevelConfigurable="true"
  IsFileCollectable="true">
 <dp:TraceComponentList>
 <dp:TraceComponent Name="agi" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="ccag" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="dba" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="dbw" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="mds" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="nm" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="nmm" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="nms" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="rtr" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 <dp:TraceComponent Name="rts" Description="ICM Process for Component RouterA"
IsLevelConfigurable="true"
  IsFileCollectable="true" />
 </dp:TraceComponentList>
</dp:TraceComponent>
<dp:TraceComponent Name="Cisco ICM Diagnostic Framework" Description="Cisco ICM Diagnostic
 Framework"
```

```
  IsLevelConfigurable="true" IsFileCollectable="true" />
<dp:TraceComponent Name="Web Setup" Description="Web Setup" IsLevelConfigurable="true"
IsFileCollectable="true" />
</dp:TraceComponentList>
</dp:ListTraceComponentsReply>
```

ListTraceFiles

Lists trace files for that application component/subcomponent during the FromDate and ToDate parameters (which are in UTC). Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/ListTraceFiles?Component/
Subcomponent&FromDate=0&ToDate=0&UseTzadjustoff=NO&Random=1467902083597

Reply example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dp:ListTraceFilesReply ReturnCode="0"
>
<dp:Schema Version="1.0"/>
<dp:TraceFileList>
<dp:FileProperty Name="TraceFile1.TXT" Date="1212347735" Size="1000000"/>
<dp:FileProperty Name="TraceFile2.TXT" Date="1212347835" Size="1000000"/>
<dp:FileProperty Name="TraceFile3.TXT" Date="1212347935" Size="1000000"/>
</dp:TraceFileList>
</dp:ListTraceFilesReply>
```

Note Optional URL parameter Type is applicable only for components that generate multiple trace types.

Note URL parameters FromDate and ToDate are used to specify time range of trace files requested by user. Unified ICM components must supply these parameters.

Note By default value entered in the field **UseTzadjustoff** is **NO**. Set the **UseTzadjustoff** to **YES**, only if the user is gathering logs across a **Daylight Savings Time**(DST) change.

Note Attribute “Date” specifies file modification time in UTC.

Note Attribute “Size” specifies file size in bytes.

DownloadTraceFile

Download the trace files that were returned by the ListTraceFiles API.

Note Only one file may be requested at a time.

However, for trace files, the ListTraceFiles API returns one zip file (including trace files, capture files, and others). You need only one download request.

Note Subsequent download requests with the same filename return with an error because after the file is downloaded, it is deleted from the server.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/DownloadTraceFile?Component= Component/
Subcomponent&File=TraceFile1.txt
```

Reply:

There are four possible replies:

- The server streams the specified file unzipped over the existing HTTP connection. Content (MIME) type is defined by the app server as “application/text”.
- The server streams the specified file zipped over the existing HTTP connection. Content (MIME) type is defined by app server as “application/zip”.
- The server streams the specified file gzipped over the existing HTTP connection. Content (MIME) type is defined by app server as “application/x-gzip”.
- In case of error, app server replies error condition in following XML format (MIME type “application/xml”):

```
<?xml version="1.0" encoding="UTF-8" ?>
  <dp:DownloadTraceFileReply ReturnCode="1" ErrorString="File TraceFile1.txt not found."/>
"xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
```

ListLogComponents

Lists all possible application components that produce log files. Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/ListLogComponents
```

Reply example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dp:ListLogComponentsReply xmlns:dp="http://www.cisco.com/vtg/diagnosticportal"
ReturnCode="0">
<dp:Schema Version="1.0"/>
<dp:LogComponentList>
<dp:LogComponent Description="ICM Installation and Upgrade logs" Name="ICM Installation and
 Upgrade"/>
<dp:LogComponent Description="ICM DBA logs" Name="ICMDBA"/>
<dp:LogComponent Description="Performance Counter Logs" Name="Performance Counter"/>
<dp:LogComponent Description="Logs for troubleshooting Active Directory issues." Name="Active
 Directory"/>
<dp:LogComponent Description="Cisco ICM Diagnostic Framework Install Logs" Name="Cisco ICM
 Diagnostic Framework Install"/>
```

```
<dp:LogComponent Description="Unified System CLI Logs" Name="Unified System CLI"/>
<dp:LogComponent Description="Web Setup logs" Name="Web Setup"/>
<dp:LogComponent Description="Web Setup troubleshooting and audit trail logs" Name="Web
Setup Trail"/>
<dp:LogComponent Description="Tomcat troubleshooting logs" Name="Tomcat"/>
<dp:LogComponent Description="Windows event logs" Name="EventLog"/>
</dp:LogComponentList>
</dp:ListLogComponentsReply>
```

ListLogFiles

Lists log files for that application component/subcomponent during the FromDate and ToDate parameters (which are in UTC). Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/ListLogFiles?Component=
Component/Subcomponent&FromDate=0&ToDate=0

Reply example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dp:ListLogFilesReply ReturnCode="0">
<dp:Schema Version="1.0"/>
<dp:LogFileList>
 <dp:FileProperty Name="LogFile1.txt" Date="1212347735" Size="1000000"/>
 <dp:FileProperty Name="LogFile2.txt" Date="1212347835" Size="1000000"/>
 <dp:FileProperty Name="LogFile3.txt" Date="1212347935" Size="1000000"/>
</dp:LogFileList>
</dp:ListLogFilesReply>
```

DownloadLogFile

Download the log files that were returned by the ListLogFiles API.

Note Only one file may be requested at a time.

In the case of downloading the log files, a user may request a subsequent download with the same filename, and the exact same file is returned. This is different from the trace file because we are not deleting the log file from the server.

Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/DownloadLogFile?Component=Component/Subcomponent&File=LogFile1.txt

Reply:

There are four possible replies:

- The server streams the specified file unzipped over the existing HTTP connection. Content (MIME) type is defined by the app server as “application/text”.
- The server streams the specified file zipped over the existing HTTP connection. Content (MIME) type is defined by app server as “application/zip”.
- The server streams the specified file zipped over the existing HTTP connection. Content (MIME) type is defined by app server as “application/x-gzip”.

- In case of error, app server replies error condition in following XML format (MIME type “application/xml”):

```
<?xml version="1.0" encoding="UTF-8" ?>
<dp:DownloadLogFileReply ReturnCode="1" ErrorString="File LogFile1.txt not found."
xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
```

ListAppServers

Lists the applications and application components installed on the target server. Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/ListAppServers

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:ListAppServersReply ReturnCode="0"
>
 <dp:Schema Version="1.0" />
 <dp:AppServerList>
 <dp:AppServer Name="buzzards-bay" ProductType="ICM"
 ProductComponentType="Logger A" />
 <dp:AppServer Name="buzzards-bay" ProductType="ICM"
 ProductComponentType="Router A" />
 <dp:AppServer Name="buzzards-bay" ProductType="ICM"
 ProductComponentType="Cisco ICM Diagnostic Framework" />
 </dp:AppServerList>
 </dp:ListAppServersReply>
```

<AppServer> has following optional attributes:

- ProductType: for product to reply topology information. Must be one of the following (“CVP”, “Unified CCX”, “Unified CM”, “Unified CCE”, “EA”, “Cisco IOS Firewall”).
- ProductComponentType: component type within a product. For example: “Router”, “PG”, and so on.

ListConfigurationCategories

Lists the configuration categories available on this application server. Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/ListConfigurationCategories

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:ListConfigurationCategoriesReply ReturnCode="0">
 <dp:Schema Version="1.0" />
 <dp:ConfigurationCategoryList>
 <dp:ConfigurationCategory Name="DumpCfg" Description="ConfigurationCategory
 for DumpCfg; Instance=acme" />
 <dp:ConfigurationCategory Name="ExportICMCfg"
 Description="ConfigurationCategory for ExportICMCfg; Instance=acme" />
 <dp:ConfigurationCategory Name="ConfigExport"
 Description="ConfigurationCategory for ConfigExport; Instance=acme" />
 <dp:ConfigurationCategory Name="Registry" Description="ConfigurationCategory
 for Registry; Instance=acme" />
 </dp:ConfigurationCategoryList>
 </dp:ListConfigurationCategoriesReply>
```

GetConfigurationCategory

Retrieve configuration information based on category. Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/GetConfigurationCategory?Category=<*category*>

Categories are: “DumpCfg”, “ExportICMCfg”, “ConfigExport”, and “Registry”.

Reply example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dp:GetConfigurationCategoryReply ReturnCode="0">
<dp:Schema Version="1.0"/>
</dp:GetConfigurationCategoryReply>
```

The requested configuration data is returned as a zip file.

GetProductVersion

Fetches the version of the applications installed on the target server.

Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/GetProductVersion

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:GetProductVersionReply ReturnCode="0">
<dp:Schema Version="1.0" />
<dp:ProductVersion Name="ICM" Major="10" Minor="0" Maintenance="1"
 VersionString="10.0(1) BuildNumber=4120" />
</dp:GetProductVersionReply>
```

GetProductLicense

Get license information for applications installed on target server.

Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/GetProductLicense

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:GetProductLicenseReply ReturnCode="0"
xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:LicenseList>
 <dp:License>
 <dp:PropertyList>
 <dp:Property Name="License" Value="Unified ICM/Unified CCE does not have any license
information." />
 </dp:PropertyList>
 </dp:License>
</dp:LicenseList>
</dp:GetProductLicenseReply>
```

GetNetStat

Run a NETSTAT command remotely on the target server and return the results.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetNetStat?Arguments="-an"
```

Reply:

Returns a text file with the output from the command that was run.

GetIPConfig

Run an IPCONFIG command remotely on the target server and return the results.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetIPConfig?Arguments="/all"
```

Reply:

Returns a text file with the output from the command that was run.

GetTraceRoute

Run a TRACERT command remotely on the target server and return the results.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetTraceRoute
```

Reply:

Returns a text file with the output from the command that was run.

GetPing

Run a PING command remotely on the target server and return the results.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetPing?Arguments="n.n.n.n"
```

Reply:

Returns a text file with the output from the command that was run.

ListProcesses

Lists application processes running on the target server.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/ListProcesses
```

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:ListProcessesReply ReturnCode="0" xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:ServiceList>
 <dp:Service Name="Logger A">
 <dp:ProcessList>
 <dp:ProcessProp Name="nodeman.exe" Description="nodeman" />
 <dp:ProcessProp Name="nmm.exe" Description="nmm" />
 <dp:ProcessProp Name="configlogger.exe" Description="configlogger" />
 <dp:ProcessProp Name="csfs.exe" Description="csfs" />
 <dp:ProcessProp Name="cw2kfeed.exe" Description="cw2kfeed" />
 <dp:ProcessProp Name="histlogger.exe" Description="histlogger" />
 <dp:ProcessProp Name="recovery.exe" Description="recovery" />
 <dp:ProcessProp Name="replication.exe" Description="replication" />
 </dp:ProcessList>
 </dp:Service>
 <dp:Service Name="Router A">
 <dp:ProcessList>
 <dp:ProcessProp Name="nodeman.exe" Description="nodeman" />
 <dp:ProcessProp Name="nmm.exe" Description="nmm" />
 <dp:ProcessProp Name="ccagent.exe" Description="ccagent" />
 <dp:ProcessProp Name="dbagent.exe" Description="dbagent" />
 <dp:ProcessProp Name="mdsproc.exe" Description="mdsproc" />
 <dp:ProcessProp Name="router.exe" Description="router" />
 <dp:ProcessProp Name="rtsvr.exe" Description="rtsvr" />
 <dp:ProcessProp Name="testsync.exe" Description="testsync" />
 </dp:ProcessList>
 </dp:Service>
 <dp:Service Name="Cisco ICM Diagnostic Framework">
 <dp:ProcessList>
 <dp:ProcessProp Name="DiagFwSvc.exe" Description="DiagFwSvc" />
 </dp:ProcessList>
 </dp:Service>
 </dp:ServiceList>
 </dp:ListProcessesReply>
```

ListServices

Lists application services running on the target server.

Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/ListServices

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:ListServicesReply ReturnCode="0"
http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:ServiceList>
 <dp:Service Name="Cisco ICM acme LoggerA" Description="Cisco ICM acme
 LoggerA" Status="Running" StartupType="Auto"
 LogOnAs="SILVERBACK.CISCO.COM\ACME-LOGGERA-77B585" />
 <dp:Service Name="Cisco ICM acme RouterA" Description="Cisco ICM acme
 RouterA" Status="Running" StartupType="Auto" LogOnAs="LocalSystem" />
 <dp:Service Name="Cisco ICM Diagnostic Framework" Description="Provides a
 web-based diagnostic service for Cisco Unified ICM,
 Enterprise application." Status="Running" StartupType="Auto"
 LogOnAs="silverback\w2008admin" />
```

```
</dp:ServiceList>
</dp:ListServicesReply>
```

GetPerformanceInformation

Get a set of System and Application Performance Counters for the specified server.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetPerformanceInformation?Component=
Component/Subcomponent
```

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:GetPerformanceInformationReply ReturnCode="0"
xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:PerformanceInformation>
<dp:PropertyList>
 <dp:Property Name="Memory/Memory Page Faults/sec" Value="29.93962" />
 <dp:Property Name="Process(_Total)/Handle Count" Value="20386" />
 <dp:Property Name="Processor(_Total)/% Processor Time" Value="13.63913" />
 <dp:Property Name="Memory/Total Memory" Value="1.399697E+09" />
 <dp:Property Name="System/Threads" Value="1165" />
 <dp:Property Name="Memory/Memory Pages/Sec" Value="3.654335" />
 <dp:Property Name="System/Processor Queue" Value="0" />
 <dp:Property Name="System/Processes" Value="73" />
 <dp:Property Name="Cisco ICM Logger(acme LoggerA)/DB Write Average Time" Value="0" />
 <dp:Property Name="Cisco ICM Logger(acme LoggerA)/DB Write Records processed" Value="0"
/>
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Calls/sec" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Agents Logged On" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Calls In Progress" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Calls In Queue" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Router State Size(KB)" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Messages Processed/sec" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Bytes Processed/sec" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Avg Process Time/Message (ms)" Value="0"
 />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Max Process Time(ms)" Value="0" />
 <dp:Property Name="Cisco ICM Router(acme RouterA)/Calls In Router" Value="0" />
</dp:PropertyList>
</dp:PerformanceInformation>
</dp:GetPerformanceInformationReply>
```

GetPerfCounterValue

Get the current value of a performance counter from the target server.

Request

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/GetPerfCounterValue?
CategoryName=Processor&CounterName="%Processor Time"&PerfInstance="_Total"
```

Reply Example

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:GetPerfCounterValueReply ReturnCode="0"
xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:PerformanceInformation>
<dp:PropertyList>
 <dp:Property Name="CategoryName" Value="Processor" />
 <dp:Property Name="CounterName" Value="% Processor Time" />
 <dp:Property Name="InstanceName" Value="_Total" />
 <dp:Property Name="BaseValue" Value="0" />
 <dp:Property Name="CounterFrequency" Value="0" />
 <dp:Property Name="CounterTimeStamp" Value="0" />
 <dp:Property Name="CounterType" Value="Timer100NsInverse" />
 <dp:Property Name="RawValue" Value="203276171875" />
 <dp:Property Name="NextValue" Value="0.003199898" />
 <dp:Property Name="SystemFrequency" Value="2333380000" />
 <dp:Property Name="TimeStamp" Value="48917923479390" />
 <dp:Property Name="TimeStamp100nSec" Value="128929442042854145" />
</dp:PropertyList>
</dp:PerformanceInformation>
</dp:GetPerfCounterValueReply>
```

GetAlarms

Retrieves up to 25 of the most recent alarms generated by the Unified CCE.

Request:

https://<*server*>:<*port*>/icm-dp/rest/DiagnosticPortal/GetAlarms?Severity=
#?Count=##

Severity and Count are optional parameters. Severity may be a numeric value between “1” and “3” (“1”=Informational, “2”=Warning, “3”=Error). Severity returns all alarms with a severity greater-than or equal-to the specified severity. Count may be a numeric value between “1” and “25”. Count returns a maximum of the specified number of alarms.

Reply example:

```
<?xml version="1.0" encoding="utf-8" ?>
<dp:GetAlarmsReply ReturnCode="0" xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">
<dp:Schema Version="1.0" />
<dp:AlarmList>
 <dp:Alarm DateTime="Jul 24, 2009 15:41:41 +0000" Type="Clear" Id="1028104" Severity="1"
Instance="acme"
Component="4_5_BERKSHIRE_ICM\acme\LoggerB" SubComponent="nm" Message="ICM\acme\LoggerB Node
 Manager started. Last
shutdown was due to system shutdown." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:27 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_hlgr" SubComponent="rtr" Message="Side B hlgr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:37 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_clgr" SubComponent="rtr" Message="Side B clgr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:27 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_clgr" SubComponent="rtr" Message="Side B clgr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:14 +0000" Type="Clear" Id="10F8004" Severity="1"
```

```
Instance="acme"
Component="6_1_BERKSHIRE_B_PG01" SubComponent="ccag" Message="Device PG01 path changing to
 idle state." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:14 +0000" Type="Clear" Id="102C107" Severity="1"
Instance="acme"
Component="4_1_BERKSHIRE_ICM\acme\RouterB" SubComponent="nm" Message="ICM\acme\RouterB Node
 Manager started. Last
shutdown was for reboot afterfailure of critical process." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:13 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_rts" SubComponent="rtr" Message="Side B rts process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:12 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_rtr" SubComponent="rtr" Message="Side B rtr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:12 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_tsyr" SubComponent="rtr" Message="Side B tsyr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:12 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_csfs" SubComponent="rtr" Message="Side B csfs process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:12 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_rcv" SubComponent="rtr" Message="Side B rcv process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:41:12 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_dba" SubComponent="rtr" Message="Side B dba process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:20 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_rtr" SubComponent="rtr" Message="Side B rtr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:20 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_tsyr" SubComponent="rtr" Message="Side B tsyr process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:20 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_csfs" SubComponent="rtr" Message="Side B csfs process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:20 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_rcv" SubComponent="rtr" Message="Side B rcv process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:20 +0000" Type="Clear" Id="10500FF" Severity="1"
Instance="acme"
Component="24_1_B_dba" SubComponent="rtr" Message="Side B dba process is OK." />
 <dp:Alarm DateTime="Jul 24, 2009 15:42:18 +0000" Type="Clear" Id="1040023" Severity="1"
Instance="acme"
Component="5_1_0" SubComponent="mds" Message="Communication with peer Synchronizer
established." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:55 +0000" Type="Clear" Id="1028103" Severity="1"
Instance="acme"
Component="4_4_WACHUSETT_ICM\acme\Distributor" SubComponent="nm" Message="ICM\acme\Distributor
 Node Manager
started. Last shutdown was by operator request." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:41 +0000" Type="Clear" Id="102C110" Severity="2"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_uaw" SubComponent="nm"
Message="ICM\acme\Distributor node process
uaw successfully reinitialized after restart." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:40 +0000" Type="Clear" Id="102C10A" Severity="2"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_uaw" SubComponent="nm"
Message="ICM\acme\Distributor node restarting
process uaw after having delayed restart for 1 seconds." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:39 +0000" Type="Raise" Id="102C10F" Severity="2"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_uaw" SubComponent="nm" Message="Process uaw
on ICM\acme\Distributor
```

```
is down after running for 30 seconds. It will restart after
        delaying 1 second for related operations to complete." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:39 +0000" Type="Raise" Id="102C10E" Severity="3"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_uaw" SubComponent="nm" Message="Process uaw
on ICM\acme\Distributor
went down for unknown reason. Exit code 0x1. It will be
        automatically restarted." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:14 +0000" Type="Clear" Id="102C111" Severity="1"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_rpl" SubComponent="nm"
Message="ICM\acme\Distributor node process
rpl successfully started." />
 <dp:Alarm DateTime="Jul 24, 2009 15:37:13 +0000" Type="Clear" Id="102C111" Severity="1"
Instance="acme"
Component="3_4_WACHUSETT_ICM\acme\Distributor_rtc" SubComponent="nm"
Message="ICM\acme\Distributor node process
rtc successfully started." />
</dp:AlarmList>
</dp:GetAlarmsReply>
```

SetAlarms

Turns the Unified CCE alarming OFF or ON. Turning the alarming OFF is useful during maintenance windows to prevent flooding at the management station.

Request:

```
https://<server>:<port>/icm-dp/rest/DiagnosticPortal/SetAlarms?State=ON|OFF
```

Reply example:

<?xml version="1.0" encoding="utf-8" ?>

<dp:SetAlarmsReply ReturnCode="0"

>

<dp:Schema Version="1.0" />

</dp:SetAlarmsReply>

SNMP/Syslog REST API

The Unified CCE SNMP implementation includes a set of SNMP agents (one primary and a set of subagents), and a service that manages the agent infrastructure. The SNMP primary agent is configured using a Microsoft Management Console (MMC) snap-in application that provides a simple user interface.

This user interface does the following:

- configuration parameters from the user
- saves the parameters
- signals the management service to restart the agents

These tasks can also be accomplished using the REST APIs for Community, User, and Traps properties.

General Information

The General Information API allows you to configure system level information that can be fetched by any NMS that query a particular instrumentation. It also adds restrictions to various sub agents that can be loaded by the SNMP primary agent.

The default port to get any instrumentation detail is 161. You can change this port by using the General Information API. You can also use the General Information API to set the SNMP log trace level.

Both SNMP and SysLog use some part of the General information API. For more information, see *Serviceability Guide for Cisco Unified ICM/Contact Center Enterprise*.

Get

This implementation of the SNMP GeneralInfo string retrieves the details of the General Information API.

Table 9: Parameters of General Information API String Get

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/GeneralInfo`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint retrieves the details of the SNMP General Information. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/GeneralInfo`
Content-type	Application/XML
Accept	Application/XML
Request Data Structure	In this implementation, the user sends the identifier in the request path parameter.

Response Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`<dp:GeneralInformationReply ReturnCode="0">`
	`   <dp:Schema Version="1.0"/>`
	`   <dp:GeneralInformation>`
	`   <dp:systemName>`*`user_name`*`</dp:systemName>`
	`   <dp:systemLocation>----</dp:systemLocation>`
	`   <dp:systemContact>----</dp:systemContact>`
	`   <dp:systemDescription>Cisco Contact Center Application Server</dp:systemDescription>`
	
	`   <dp:agentPollsPort>1</dp:agentPollsPort>`
	`   <dp:enableAuthenticationTraps>false</dp:enableAuthenticationTraps>`
	`   <dp:agentExecutionPriority>1</dp:agentExecutionPriority>`
	`   <dp:maximumConcurrentRequests>5</dp:maximumConcurrentRequests>`
	`   <dp:maximumSubagentWaitTime>25</dp:maximumSubagentWaitTime>`
	`   <dp:maximumSubagents>25</dp:maximumSubagents>`
	`   <dp:agentLogQuantity>1</dp:agentLogQuantity>`
	`   </dp:GeneralInformation>`
	`</dp:GeneralInformationReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value
systemName	Represents the host name (or fully qualified domain name).	String	Length <= 127	No	Alphanumeric
systemLocation	Represents the physical location of this host.	String	Length <= 127	No	Alphanumeric
systemContact	Represents the name of the person to contact if a problem arises with this host.	String	Length <= 127	No	Alphanumeric
systemDescription	Represents the Description of this host(any information deemed relevant).	String	Length <= 127	No	Alphanumeric
agentPollsPort	Represents the port number that the SNMP primary agent listens for inbound requests.	String	Length <= 5	No	Numeric
enableAuthenticationTraps	When enabled, sends a trap when an authentication failure occurs.	Boolean	-	No	true /false; case sensitive
agentExecutionPriority	Represents the thread execution priority for all SNMP agents.	Int	0-2	No	-
maximumConcurrentRequests	Represents the maximum concurrent SNMP Objects request that are allowed.	Unsigned Int	2-32	No	-
maximumSubagentWaitTime	Represents the maximum number of seconds a primary agent will wait for a subagent to respond to a request (timeout).	Unsigned Int	5-150	No	
maximumSubagents	Represents the maximum number of SNMP subagents the primary agent permits to be loaded.	Unsigned Int	5-150	No	
agentLogQuantity	Represents the number of log messages to be written to the agent log files.	Int	0-2	No	

Update

This implementation of the SNMP GeneralInfo string updates the properties of the General Information API.

Table 10: Parameters of General Information API String Update

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/GeneralInfo`
HTTP Method	PUT
Input/Output Format	XML
Request	The PUT operation on the service endpoint updates the properties of the General Information API. `PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/GeneralInfo`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<GeneralInformation>`
	  `<systemName>MachineName.MachineDomain.instance</systemName>`
	  `<systemLocation>-</systemLocation>`
	  `<systemContact>-</systemContact>`
	  `<systemDescription>Cisco Contact Center Application Server</systemDescription>`
	  `<agentPollsPort>1</agentPollsPort>`
	  `<enableAuthenticationTraps>false</enableAuthenticationTraps>`
	  `<agentExecutionPriority>1</agentExecutionPriority>`
	  `<maximumConcurrentRequests>5</maximumConcurrentRequests>`
	  `<maximumSubagentWaitTime>25</maximumSubagentWaitTime>`
	  `<maximumSubagents>25</maximumSubagents>`
	  `<agentLogQuantity>2</agentLogQuantity>`
	`</GeneralInformation>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value
systemName	Represents the host name (or fully qualified domain name).	String	Length <= 127	No	Alphanumeric

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value
systemLocation	Represents the physical location of this host.	String	Length <= 127	No	Alphanumeric
systemContact	Represents the name of the person to contact if a problem arises with this host.	String	Length <= 127	No	Alphanumeric
systemDescription	Represents the Description of this host(any information deemed relevant).	String	Length <= 127	No	Alphanumeric
agentPollsPort	Represents the port number that the SNMP primary agent listens for inbound requests.	String	Length <= 5	No	-
enableAuthenticationTraps	When enabled, sends a trap when an authentication failure occurs.	Boolean	-	No	true/false; case sensitive
agentExecutionPriority	Represents the thread execution priority for all SNMP agents.	Int	0-2	No	-
maximumConcurrentRequests	Represents the maximum concurrent SNMP Objects request that are allowed.	Unsigned Int	2-32	No	-
maximumSubagentWaitTime	Represents the maximum number of seconds a primary agent will wait for a subagent to respond to a request (timeout).	Unsigned Int	5-150	No	
maximumSubagents	Represents the maximum number of SNMP subagents the primary agent permits to be loaded.	Unsigned Int	5-150	No	
agentLogQuantity	Represents the number of log messages to be written to the agent log files.	Int	0-2	No	

SNMP v1/v2c Community

If you are using SNMP v1 or v2c you must configure a community name so that Network Management Stations (NMS) can access the data provided by your server. These names are left blank during installation for security reasons.

SNMP community names are used to authenticate data exchange of SNMP information. An NMS can exchange SNMP information only with servers that use the same community name.

Note SNMP community name along with the SNMP Version forms an unique entity and acts as a primary key.

Create

This implementation of SNMP v1/v2c community string creates a SNMP community mentioned in the request.

Table 11: Parameters of SNMP v1/v2c Community String Create

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`
HTTP Method	POST
Input/Output Format	XML
Request	The POST operation on the service endpoint creates the SNMP Community. `POST https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`<community>`
	`  <name>Public_Community</name>`
	`  <snmpversion>V1</snmpversion>`
	`  <accessprivilege>ReadOnly</accessprivilege>`
	`  <hosts>`
	`    <host>192.0.2.0</host>`
	`  </hosts>`
	`</community>`

Response Header	Return 201 CREATED.
	`Response headers:`
	`HTTP/1.1 201 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`
	`Date: Mon, 26 Aug 2013 09:15:27 GMT`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the community string name	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
snmpversion	Represents the SNMP version information	String	-	No	V1 (default), V2c	Only V1 or V2c; Not case sensitive.
hosts	Represents a list of "host" tags	-	-	No	If this tag is specified, then the host tag is mandatory.	-
host	Represents the management station host IP address. If the management station host IP address is not provided then any host can request for the management information.	String	Valid IP address; number of host tags must be <= 255	No	A valid ip4 address	-
accessprivilege	Represents the access privileges of the community string.	String	-	No	ReadOnly (default), ReadWrite	Only ReadOnly or ReadWrite; Not case sensitive.

Delete

This implementation of SNMP v1/v2c community string deletes the community from the listed devices.

Table 12: Parameters of SNMP v1/v2c Community String Delete

URL	DELETE `https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community/Name/<name>&SnmpVersion/<snmpversion>`
HTTP Method	DELETE
Input/Output Format	XML
Request	The DELETE operation on the service endpoint deletes the SNMP Community. `DELETE https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community/Name/<name>&SnmpVersion/<snmpversion>`
Content-type	Application/XML
Accept	Application/XML
Request Data Structure	In this implementation, the user sends the identifier in the request path parameter.
Response Header	Return 200 OK. `Response headers:` `HTTP/1.1 200 OK` `Transfer-Encoding: chunked` `Content-Type: application/xml` `Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Get

This implementation of the SNMP v1/v2c community string retrieves the details of the community.

Table 13: Parameters of SNMP v1/v2c Community String Get

URL	GET `https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community/Name/<name>&SnmpVersion/<snmpversion>`
HTTP Method	GET
Input/Output Format	XML

Request	The GET operation on the service endpoint retrieves the details of the SNMP community.
	`GET`
	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community/Name/<name>&SnmpVersion/<snmpversion>`

Content-type	Application/XML
Accept	Application/XML
Request Data Structure	In this implementation, the user sends the identifier in the request path parameter.

Response Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`  <dp:CommunityReply ReturnCode="0" xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">`
	`    <dp:Schema Version="1.0" />`
	`    <dp:community>`
	`      <dp:name>str1</dp:name>`
	`      <dp:snmpversion>V2c</dp:snmpversion>`
	`      <dp:accessprivilege>ReadOnly</dp:accessprivilege>`
	`      <dp:hosts>`
	`        <dp:host>192.0.2.0</dp:host>`
	`        <dp:host>192.0.2.1</dp:host>`
	`      </dp:hosts>`
	`    </dp:community>`
	`  </dp:CommunityReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

List

This implementation of SNMP v1/v2c community string lists all the communities that are configured on the system.

Table 14: Parameters of SNMP v1/v2c Community String List

URL	**`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`**
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint lists the details of the SNMP community. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`  <dp:CommunityReply ReturnCode="0" xmlns:dp="http://www.cisco.com/vtg/diagnosticportal">`
	
	`      <dp:Schema Version="1.0" />`
	`      <dp:communities>`
	`         <dp:community>`
	`           <dp:name>george</dp:name>`
	`           <dp:snmpversion>V1</dp:snmpversion>`
	`         </dp:community>`
	`         <dp:community>`
	`           <dp:name>public_community</dp:name>`
	`           <dp:snmpversion>V2c</dp:snmpversion>`
	`         </dp:community>`
	`      </dp:communities>`
	`  </dp:CommunityReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the community string name	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.
snmpversion	Represents the SNMP version information	String	-	No	V1 (default), V2c	Only V1 or V2c; Not case sensitive.

Update

This implementation of SNMP v1/v2c community string updates the properties of the SNMP community string.

Only full update of SNMP v1/v2c Community API is allowed. Therefore, all the XML tags with valid values must be included while sending the update request. For more information, see Update Implementation for SNMP/Syslog REST APIs, on page 89.

Table 15: Parameters of SNMP v1/v2c Community String Update

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`
HTTP Method	PUT
Input/Output Format	XML
Request	The PUT operation on the service endpoint updates the properties of the SNMP community. `PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`<community>`
	`    <name>Public_Community</name>`
	`    <snmpversion>V1</snmpversion>`
	`    <accessprivilege>ReadOnly</accessprivilege>`
	`    <hosts>`
	`       <host>192.0.2.0</host>`
	`    </hosts>`
	`</community>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error (for example, the connection is broken with the database server or ORM or any other component. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the community string name	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.
snmpversion	Represents the SNMP version information	String	-	Yes	V1 (default), V2c	Only V1 or V2c; Not case sensitive.
hosts	Represents a list of "host" tags	-	-	Yes	If this tag is specified, then the host tag is mandatory.	-
host	Represents the management station host IP address. If the management station host IP address is not provided then any host can request for the management information.	String	Valid IP address; number of host tags must be <= 255	Yes	A valid ip4 address	-
accessprivilege	Represents the access privileges of the community string.	String	-	Yes	ReadOnly (default), ReadWrite	Only ReadOnly or ReadWrite; Not case sensitive.

SNMPv3 User

If you are using Simple Network Management Protocol Version 3 (SNMPv3) you must configure a user name so that the Network Management Stations (NMS) can access the data provided by your server.

Create

This implementation of SNMP user string creates an SNMP user as mentioned in the request.

Table 16: Parameters of SNMP User String Create

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`
HTTP Method	POST
Input/Output Format	XML
Request	The POST operation on the service endpoint creates the SNMP User. `POST https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<user>`
	`   <name>Snmp_User</name>`
	`   <accessprivilege>ReadOnly</accessprivilege>`
	`   <hosts>`
	`       <host>192.0.2.0</host>`
	`   </hosts>`
	`   <authInfoReqd>true</authInfoReqd>`
	`   <authProtocol>MD5</authProtocol>`
	`   <authPassword>user@123</authPassword>`
	`   <privacyInfoReqd>true</privacyInfoReqd>`
	`   <privacyProtocol>AES-192</privacyProtocol>`
	`   <privacyPassword>user@123456</privacyPassword>`
	`</user>`

Response Header	Return 201 CREATED.
	`Response headers:`
	`HTTP/1.1 201 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`
	`Date: Mon, 26 Aug 2013 09:15:27 GMT`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the user string name.	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.
accessprivilege	Represents the access privileges of the user string.	String	-	No	ReadOnly (default), ReadWrite	Only ReadOnly or ReadWrite; Not case sensitive.
hosts	Represents a list of "host" tags.	-	-	No	If this tag is specified, then the host tag is mandatory.	-
host	Represents the management station host IP address. If the management station host IP address is not provided, then any host can request for the management information.	String	Valid IP address; number of host tags must be <= 255	No	A valid ip4 address	-
authInfoReqd	Authentication information is required.	Boolean	-	No. If this tag is not provided, then the default value, false, will be considered.	false, true	Case sensitive.
authProtocol	Authentication protocol.	String	-	No. If authInfoReqd is true, then this field is mandatory.	MD5 SHA-1	Value for this tag is set only when authInfoReqd is true.
authPassword	Authentication password.	String	-	No. If authInfoReqd is true, then this field is mandatory.	Alphanumeric with any special character.	Value for this tag is set only when authInfoReqd is true.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
privacyInfoReqd	Privacy information is required.	Boolean	-	No.	false, true	Case sensitive. If privacyInfoReqd is true, then authInfoReqd must be set to true.
privacyProtocol	Privacy protocol.	String	-	No. If privacyInfoReqd is true, then this field is mandatory.	3DES AES-192 AES-256	Value for this tag is set only when privacyInfoReqd is true.
privacyPassword	Privacy password.	String	-	No. If privacyInfoReqd is true, then this field is mandatory.	Alphanumeric with any special character.	Value for this tag is set only when privacyInfoReqd is true.

Delete

This implementation of SNMP user string deletes the user from the listed devices.

Table 17: Parameters of SNMP User String Delete

URL	DELETE `https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user/Name/<name>`
HTTP Method	DELETE
Input/Output Format	XML
Request	The DELETE operation on the service endpoint deletes the SNMP User. `DELETE https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user/Name/<name>`
Content-type	Application/XML
Accept	Application/XML
Request Data Structure	In this implementation, the user sends the identifier in the request path parameter.

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Get

This implementation of the SNMP user string retrieves the details of the user.

Table 18: Parameters of SNMP User String Get

URL	GET `https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user/Name/<name>`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint retrieves the details of the SNMP User. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user/Name/<name>`
Content-type	Application/XML
Accept	Application/XML
Request Data Structure	In this implementation, the user sends the identifier in the request path parameter.

Response Data Structure	`<dp:UserReply ReturnCode="0">`
	   `<dp:Schema Version="1.0"/>`
	      `<dp:user>`
	          `<dp:name>Snmp_User</dp:name>`
	          `<dp:accessprivilege>ReadOnly</dp:accessprivilege>`
	      `<dp:hosts>`
	          `<dp:host>192.0.2.0</dp:host>`
	      `</dp:hosts>`
	      `<dp:authInfoReqd>true</dp:authInfoReqd>`
	      `<dp:authProtocol>MD5</dp:authProtocol>`
	      `<dp:privacyInfoReqd>true</dp:privacyInfoReqd>`
	      `<dp:privacyProtocol>AES-192</dp:privacyProtocol>`
	      `</dp:user>`
	`</dp:UserReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

List

This implementation of SNMP User string lists all the users that are configured on the system.

Table 19: Parameters of SNMP User String List

URL	**`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`**
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint lists the details of the SNMP User. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<dp:ListUserReply ReturnCode="0">`
	   `<dp:Schema Version="1.0"/>`
	       `<dp:users>`
	            `<dp:name>Snmp_User1</dp:name>`
	            `<dp:name>Snmp_User2</dp:name>`
	            `<dp:name>Snmp_User3</dp:name>`
	       `</dp:users>`
	`</dp:ListUserReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the user string name.	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.

Update

This implementation of user string updates the properties of the SNMP user string.

Only full update of SNMP v3 User API is allowed. Therefore, all the XML tags with valid values must be included while sending the update request. For more information, see Update Implementation for SNMP/Syslog REST APIs, on page 89.

Table 20: Parameters of SNMP User String Update

URL	**`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`**
HTTP Method	PUT
Input/Output Format	XML
Request	The PUT operation on the service endpoint updates the properties of the SNMP community. `PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/user`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<user>`
	`  <name>Snmp_User</name>`
	`  <accessprivilege>ReadOnly</accessprivilege>`
	`  <hosts>`
	`     <host>192.0.2.0</host>`
	`  </hosts>`
	`  <authInfoReqd>true</authInfoReqd>`
	`  <authProtocol>MD5</authProtocol>`
	`  <authPassword>user@123</authPassword>`
	`  <privacyInfoReqd>true</privacyInfoReqd>`
	`  <privacyProtocol>AES-192</privacyProtocol>`
	`  <privacyPassword>user@123456</privacyPassword>`
	`</user>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error (for example, the connection is broken with the database server or ORM or any other component). 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the user string name.	String	Length <= 40	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed. Space is not allowed.
accessprivilege	Represents the access privileges of the user string.	String	-	Yes	ReadOnly (default), ReadWrite	Only ReadOnly or ReadWrite; Not case sensitive.
hosts	Represents a list of "host" tags.	-	-	Yes	If this tag is specified, then the host tag is mandatory.	-
host	Represents the management station host IP address. If the management station host IP address is not provided, then any host can request for the management information.	String	Valid IP address; number of host tags must be <= 255	Yes	A valid ip4 address	-
authInfoReqd	Authentication information is required.	Boolean	-	Yes	false, true	Case sensitive.
authProtocol	Authentication protocol.	String	-	No. If authInfoReqd is true, then this field is mandatory.	MD5 SHA-1	Value for this tag is set only when authInfoReqd is true.
authPassword	Authentication password.	String	-	No. If authInfoReqd is true, then this field is mandatory.	Alphanumeric with any special character.	Value for this tag is set only when authInfoReqd is true.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
privacyInfoReqd	Privacy information is required.	Boolean	-	Yes	false, true	Case sensitive. If privacyInfoReqd is true, then authInfoReqd must be set to true.
privacyProtocol	Privacy protocol.	String	-	No. If privacyInfoReqd is true, then this field is mandatory.	3DES AES-192 AES-256	Value for this tag is set only when privacyInfoReqd is true.
privacyPassword	Privacy password.	String	-	No. If privacyInfoReqd is true, then this field is mandatory.	Alphanumeric with any special character.	Value for this tag is set only when privacyInfoReqd is true.

Traps

Traps are messages alerting the SNMP manager to a condition on the network. Notifications can indicate various significant events such as:

- improper user authentication
- restarts
- the closing of a connection
- loss of connection to a neighboring router

and so on.

Create

This implementation of SNMP Trap string creates trap as mentioned in the request.

Table 21: Parameters of SNMP Trap string Post

URL	`https:// <server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
HTTP Method	POST
Input/Output Format	XML
Request	The POST operation on the service endpoint creates an SNMP Trap. `POST https:// <server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
Content-type	Application/XML
Accept	Application/XML

<table>
<tr><td><b>Request Data Structure</b></td><td><pre>&lt;?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<trap>
    <name>trap1</name>
    <snmpversion>v2c</snmpversion>
    <communityOrUserRef>comm2</communityOrUserRef>
    <destinations>
         <destination>
                <ipAddr>192.0.2.0</ipAddr>
                <port>9999</port>
          </destination>
     </destinations>
</trap&gt;</pre></td></tr>
<tr><td><b>Response Header</b></td><td>Return 201 CREATED.<pre>Response headers:
HTTP/1.1 201 OK
Transfer-Encoding: chunked
Content-Type: application/xml
Server: Microsoft-HTTPAPI/2.0</pre></td></tr>
<tr><td><b>Response Code</b></td><td>400- Bad request. If the request body is invalid.
400- API error . If the object either does not exist or is stale.
403- Authorization Failure (for example, the user is not authenticated in the web session).
500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body.
503 Service Unavailable. When the request processing threshold is reached.</td></tr>
<tr><td><b>Security Constraints</b></td><td>Only a Serviceability administrator can perform this operation.</td></tr>
</table>

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the trap string name.	String	Length <= 41	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed; Space is not allowed.
snmpversion	Represents the SNMP version.	String	-	Yes	V1, V2C, V3	V1, V2C, V3
communityOrUserRef	Represents a community or user for which this trap is configured.	String	Length <= 41	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed; Space is not allowed.
destinations	Represents the list of destination tag.	-	-	Yes	-	-
destination	Each destination tag has a mandatory ipAdder and port tag.	-	Number of destination tags must be <= 255	Yes	-	-

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
ipAddr	Represents the destination IP where traps have to be sent.	String	Valid IP address	Yes	Valid IP address	-
port	Represents the port on the destination where the traps have to be sent.	Sting	Length <= 5	Yes	The default value 162 is assumed when: - no value is specified - the value specified is 0	Numeric.

Delete

This implementation of SNMP Trap string deletes the traps from the listed traps.

Table 22: Parameters of SNMP Trap string Delete

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap/Name/<name>`
HTTP Method	DELETE
Input/Output Format	XML
Request	The DELETE operation on the service endpoint deletes the properties of the Traps API. `DELETE https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap/Name/<name>`
Content-type	Application/XML
Accept	Application/XML

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Get

This implementation of SNMP Trap string retrieves the details of the trap.

Table 23: Parameters of SNMP Trap string Get

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap/Name/<name>`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint retrieves the details of the Trap API. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap/Name/<name>`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<dp:TrapReply ReturnCode="0">`
	`<dp:Schema Version="1.0"/>`
	`<dp:trap>`
	`    <dp:name>trap1</dp:name>`
	`    <dp:snmpversion>V2C</dp:snmpversion>`
	`    <dp:communityOrUserRef>comm2</dp:communityOrUserRef>`
	`    <dp:destinations>`
	`          <dp:destination>`
	`               <dp:ipAddr>192.0.2.0</dp:ipAddr>`
	`               <dp:port>9999</dp:port>`
	`           </dp:destination>`
	`     </dp:destinations>`
	`</dp:trap>`
	`</dp:TrapReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

List

This implementation of SNMP trap string lists all the traps that are configured on the system.

Table 24: Parameters of SNMP Trap string Get

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint lists all the traps that are configured on the system. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<dp:TrapReply ReturnCode="0">`
	`<dp:Schema Version="1.0"/>`
	`<dp:traps>`
	`      <dp:trap>`
	`             <dp:name>trap1</dp:name>`
	`      </dp:trap>`
	`      <dp:trap>`
	`             <dp:name>trap2</dp:name>`
	`      </dp:trap>`
	`</dp:traps>`
	`</dp:TrapReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the trap string name.	String	Length <= 41	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed.

Update

This implementation of trap string updates the properties of the SNMP trap that is mentioned in the request.

Only full update of SNMP Traps API is allowed. Therefore, all the XML tags with valid values must be present while sending the update request. For more information, see Update Implementation for SNMP/Syslog REST APIs, on page 89.

Table 25: Parameters of SNMP Trap string Put

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
HTTP Method	PUT
Input/Output Format	XML
Request	The PUT operation on the service endpoint updates the properties of the SNMP trap that is mentioned in the request. `PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/trap`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
	`<trap>`
	`    <name>trap1</name>`
	`    <snmpversion>v2c</snmpversion>`
	`    <communityOrUserRef>comm2</communityOrUserRef>`
	`    <destinations>`
	`         <destination>`
	`                <ipAddr>192.0.2.0</ipAddr>`
	`                <port>9999</port>`
	`          </destination>`
	`     </destinations>`
	`</trap>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error. If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server-side errors. Details about the error is given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
name	Represents the trap string name.	String	Length <= 41	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed.
snmpversion	Represents the SNMP version.	String	-	Yes	V1, V2C, V3	V1, V2C, V3
communityOrUserRef	Represents a community or user for which this trap is configured.	String	Length <= 41	Yes	Alphanumeric	Alphanumeric dot, underscore, and hyphens are allowed.
destinations	Represents the list of destination tag.	-	-	Yes	-	-
destination	Each destination tag has a mandatory ipAdder and port tag.	-	Number of destination tags must be <= 255	Yes	-	-
ipAddr	Represents the destination IP where traps have to be sent.	String	Valid IP address	Yes	Valid IP address	-
port	Represents the port on the destination where the traps have to be sent.	Sting	Length <= 5	Yes	The default value 162 is assumed when: - no value is specified - the value specified is 0	Numeric.

Syslog

Syslog is a method of collecting messages from devices to a server running a syslog daemon. Logging to a central syslog server helps in collection of logs and alerts. Cisco devices can send their log messages to a Unix-style SYSLOG service. A SYSLOG service simply accepts messages, and stores them in files or prints them according to a simple configuration file.

For a Syslog instance, there can a maximum of five Syslog collector destinations that you can configure to receive the Syslog messages simultaneously.

Update

This implementation of syslog updates syslog parameters that are mentioned in the request.

Only full update of Syslog API is allowed. Therefore, all the XML tags with valid values must be present while sending the update request. For more information, see Update Implementation for SNMP/Syslog REST APIs, on page 89.

Table 26: Parameters of syslog API String Create

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/syslog`
HTTP Method	PUT
Input/Output Format	XML
Request	The PUT operation on the service endpoint updates the details of the SNMP REST syslog API. `PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/syslog`
Content-type	Application/XML
Accept	Application/XML

Request Data Structure	`<?xml version="1.0" encoding="utf-8"?>`
	`<syslogParameters>`
	`  <syslogInstanceName >inst1</syslogInstanceName >`
	`  <loggerNode>LoggerA</loggerNode>`
	`  <enableFeed>true</enableFeed>`
	`  <collectorAddressList>`
	`     <collectorAddress>`
	`       <address>192.0.2.1</address>`
	`       <port>514</port>`
	`     </collectorAddress>`
	`     <collectorAddress>`
	`       <address>192.0.2.1</address>`
	`       <port>515</port>`
	`     </collectorAddress>`
	`  </collectorAddressList>`
	`  <disablePing>true</disablePing>`
	`</syslogParameters>`

Response Header	Return 200 OK.
	
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
syslogInstanceName	Represents the name of the logger instance.	String	Length <= 32	Yes	Alphanumeric	Only valid instance configured in the logger machine.

Tag	Description	Data Type	Input Constraints	Required Fields	Possible Value	Additional Validation
loggerNode	Represents the logger node.	String	-	Yes	LoggerA or LoggerB	Only LoggerA or LoggerB; Not case sensitive
enableFeed	If set to true, sends the syslog messages to the configured syslog collector. Also, any modification to the collector address and other parameters requires setting enableFeed to true.	Boolean	-	Yes	-	true/false; case sensitive.
collectorAddressList	Represents a list of collectorAddress.	-	-	Yes	-	-
collectorAddress	Each collectorAddress tag should have a mandatory address and port tag.	-	Number of collectorAddress tags must be <= 5	Yes	-	-
address	Represents the Syslog collector IP address or host name.	String	Host or IP address	Yes	Alphanumeric	A valid host name or IP address.
port	Port on the syslog collector destination.	String	Value should be <= 65535	Yes	Only numeric	514 (default). If value is specified as 0, then the default value is assumed.
disablePing	If set to true, disables ping messages to the syslog collector destination.	Boolean	-	Yes	-	true/false; case sensitive.

List

This implementation of syslog lists all the instances that are configured on the system.

Table 27: Parameters of Syslog API String List

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/syslog`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint lists the instances of the SNMP REST syslog API that are configured on the system. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/syslog`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<dp:ListLoggerInstancesReply ReturnCode="0">`
	`   <dp:Schema Version="1.0"/>`
	`   <dp:LoggerInstances>`
	`   <dp:LoggerInstance>`
	`   <dp:loggerInstanceName >inst1</dp:loggerInstanceName >`
	`   <dp:loggerNode>LoggerA</dp:loggerNode>`
	`   </dp:LoggerInstance>`
	`   </dp:LoggerInstances>`
	`</dp:ListLoggerInstancesReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Get

This implementation of syslog retrieves the details of the syslog parameters for a logger instance.

Table 28: Parameters of Syslog API String

URL	`https://<server>:<port>/icm-dp/rest/DiagnosticPortal/ syslog/syslogInstanceName /<name>&loggerNode/<node>`
HTTP Method	GET
Input/Output Format	XML
Request	The GET operation on the service endpoint retrieves the details of the syslog parameters for a logger instance. `GET https://<server>:<port>/icm-dp/rest/DiagnosticPortal/syslog/syslogInstanceName /<name>&loggerNode/<node>`
Content-type	Application/XML
Accept	Application/XML

Response Data Structure	`<dp:syslogReply ReturnCode="0">`
	`<dp:Schema Version="1.0"/>`
	`<dp:syslogParameters>`
	`<dp:syslogInstanceName>test1</dp:syslogInstanceName>`
	`<dp:loggerNode>LoggerA</dp:loggerNode>`
	`<dp:enableFeed>false</dp:enableFeed>`
	`<dp:collectorAddressList>`
	`     <dp:collectorAddress>`
	`       <dp:address>192.1.2.1</dp:address>`
	`       <dp:port>514</dp:port>`
	`     </dp:collectorAddress>`
	`     </dp:collectorAddressList>`
	`<dp:disablePing>false</dp:disablePing>`
	`</dp:syslogParameters>`
	`</dp:syslogReply>`

Response Header	Return 200 OK.
	`Response headers:`
	`HTTP/1.1 200 OK`
	`Transfer-Encoding: chunked`
	`Content-Type: application/xml`
	`Server: Microsoft-HTTPAPI/2.0`

Response Code	400- Bad request. If the request body is invalid. 400- API error . If the object either does not exist or is stale. 403- Authorization Failure (for example, the user is not authenticated in the web session). 500- Internal Server Error. This error is displayed for all generic server side errors. Details about the error will be given in the error notification that is displayed in the response body. 503 Service Unavailable. When the request processing threshold is reached.
Security Constraints	Only a Serviceability administrator can perform this operation.

Update Implementation for SNMP/Syslog REST APIs

Only full update of Community, User, Traps, and Syslog APIs is allowed. The following example demonstrates how to send an update request for the Community, User, Traps, and Syslog APIs.

Consider that a community is added with the following properties:

```
POST https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community
<?xml version="1.0" encoding="utf-8"?>
<community>
    <name>Public_Community</name>
    <snmpversion>V1</snmpversion>
    <accessprivilege>ReadOnly</accessprivilege>
    <hosts>
        <host>192.0.2.0</host>
        <host>192.0.2.1</host>
    </hosts>
</community>
```

If you want update one of the hosts from **192.0.2.0** to **192.0.2.250**, the XML body of the update request must contain the following properties:

```
PUT https://<server>:<port>/icm-dp/rest/DiagnosticPortal/snmp/community
<?xml version="1.0" encoding="utf-8"?>
<community>
    <name>Public_Community</name>
    <snmpversion>V1</snmpversion>
    <accessprivilege>ReadOnly</accessprivilege>
    <hosts>
        <host>192.0.2.250</host>             //modified IP
        <host>192.0.2.1</host>            //existing IP retained
    </hosts>
</community>
```

Diagnostic Framework Troubleshooting

The Diagnostic Framework is self contained and does not require any additional configuration other than assigning users. If you encounter any issues with the service, see the following table:

Table 29: Diagnostic Framework Troubleshooting

Issue	Troubleshooting / Remedy
Diagnostic Framework service does not start	Check if required service HTTP SSL (and IIS, when installed) is started without any errors. Check Windows Event log for errors and resolve any issues with the required services. Make sure none of the configuration files is missing. Check Event Viewer and Diagnostic Framework log file for any initialization errors.
Cannot access any API from the supported browser client	Confirm that you are using a supported browser by checking the *Contact Center Enterprise Compatibility Matrix* at https://www.cisco.com/c/en/us/support/customer-collaboration/unified-contact-center-enterprise/products-device-support-tables-list.html. Confirm the base URL is correct; compare it with the URL in the service configuration file DiagFwSvc.exe.config. Confirm the API used is valid. Make sure the API is accessed using HTTPS. Make sure the credentials used as valid, check Windows Event log for any authentication errors and Diagnostic Framework log for any authorization errors. Use DiagFwCertMgr utility to validate the certificate binding to the port in use. Recreate or rebind the certificate if any issues were found. Clear the supported browser cache and restart the browser. Verify that the Windows Firewall is either turned off, or that it was configured with the ICM Security Wizard, which ensures that a proper exception is in place for the Diagnostic Framework to work.
Some commands work, and others do not seem to work.	Confirm that you are using a supported browser by checking the *Contact Center Enterprise Compatibility Matrix* at https://www.cisco.com/c/en/us/support/customer-collaboration/unified-contact-center-enterprise/products-device-support-tables-list.html.

DUMPLOG

The DUMPLOG utility converts binary log files written by Unified ICM/Unified CCE processes into readable text format. DUMPLOG can optionally display the binary log files in Cisco Log message format. For more information about the Cisco Log format, see The Cisco Log Message Format. For more information about this utility, see the *How to Use the DumpLog Utility Tech Note* at https://www.cisco.com/en/US/products/sw/custcosw/ps1001/prod_tech_notes_list.html.

Header

Cisco Log formatted log entries include a more comprehensive header compared to DUMPLOG standard format.

DumpLog Standard Format

Standard formatted DUMPLOG entries display the following fields:

```
<TIMESTAMP> <COMPONENT-PROCESS> <MESSAGE>
```

The timestamp is represented as a 24-hour value (hh:mm:ss). It does not include the date, which appears on a separate line at the beginning of the file and when a new day starts. For example:

```
Events from February 8, 2007
```

```
00:37:44 ra-rtr MDS is in service.
```

Cisco Log Format

Cisco Log formatted DUMPLOG entries display the following fields:

```
<SEQNUM>: <HOST>: <TIMESTAMP> <TIMEZONE>: %APPNAME: %<TAGS>:<MESSAGE>
```

Example DUMPLOG message

Below is an example of a Cisco Log formatted DUMPLOG message. An actual log entry appears on a single line.

```
10: CICMRGRA: Feb 8 2007 05:37:44.658 +0000: %ICM_Router_ProcessSynchronization:
[comp=Router-A]
[pname=rtr][iid=acme][sev=info]: MDS is in service.
```

Usage

You can use the following command-line options for the **dumplog** utility to view the log files within a specific time period. You can define the time period with the /bd, /bt, /ed, and /et switches.

dumplog*[ProcessName(s)][/dir Dirs] [/if InputFile] [/o] [/of OutputFile] [/c] [/bd BeginDate(mm/dd/yyyy)] [/bt BeginTime(hh:mm:ss)] [/ed EndDate(mm/dd/yyyy)] [/et EndTime(hh:mm:ss)] [/hr HoursBack] [/all] [/last] [/prev] [bin] [/m MatchString] [/x ExcludeString] [/ms] [/debug] [/ciscoLog] [/unzipCmdPrefix Prefix for Unzip command] [/unzipCmdInfix Infix for Unzip command] [/unzipCmdPostfix Postfix for Unzip command] [/unzipTempfile Temporary filename for unzip command] [/zipPostfix Postfix of zipped files] [/tzadjustoff] [/help] [?]*

Parameter	Description
ProcessName(s)	This command dumps the current day log for this process, unless you specify different dates or times with other arguments.
[/dir Dirs]	This command specifies the directory location of the log files for any processes listed on the command line after the /dir switch. If no /dir switch is used, the current directory is used by default.
[/if]	The InputFile specifies a specific .ems file to dump. The /if token is optional. If you specify an input file, the /bd, /bt, /ed, /et, /hr, and /all arguments are ignored.
/o	Writes output to a text file in the \logfiles directory. The filename is formed when you add the .txt suffix to the specified process prefix or input file name (without the .ems suffix). The file is written to the current directory.
/of	OutputFile specifies an output text file; for example, c:\temp\mylog.txt.
/c	Specifies continuous output. The command does not exit after it reaches the end of the log. Instead, it waits and writes any further entries that appear in the log.
/bd	BeginDate(mm/dd/yyyy) specifies the begin date. If used with /bt, this specifies a range of dates. Otherwise, **dumplog** dumps events for only the specified date.
/bt	BeginTime(hh:mm:ss) specifies the begin time. Use with /et in order to specify a range of time.
/ed	EndDate(mm/dd/yyyy) specifies the end date. Use with /bd in order to specify a range of days.
/et	EndTime(hh:mm:ss) specifies the end time. Use with /bt in order to specify a range of time.
/hr	HoursBack specifies a number of hours back from the current time.
/all	Displays all information from the specified process log files.
/last	Displays information from the most recent log file for the process.
/prev	Displays information from the next to last log file for the process.
/m	MatchString displays only events that contain a match for the specified string.
/x	ExcludeString displays only events that do not contain a match for the specified string.
[/ms]	Displays milliseconds in time stamps.
[/mc]	Use multiple colors when you dump merged logs. Each process is given a different color. You must specify either a ProcessPrefix or an InputFile. If you give only a ProcessPrefix value (for example, rtr, nm, or lgr), **dumplog** displays the current day log for that process by default.
/ciscoLog	Enables the CiscoLog functionality.
/unzipCmdPrefix	Prefix parameters for unzip, for example gzip -d -c.

Parameter	Description
/unzipCmdInfix	Infix parameter for unzip, for example ">".
/unzipCmdPostfix	Postfix parameter for unzip, for example "".
/unzipTempfile	Temp file for unzip, for example "temp.ems".
/zipPostfix	File postfix parameter, for example ".gz".
/tzadjustoff	When the EMS files are copied to a system in a different timezone, or if the timezone on the system is changed, without this option, all the queries made will be relative to the machine on which the logfiles were generated. Otherwise, /tzadjustoff is used in order to switch the behavior where queries are made with respect to this machine time. **Note** Use /tzadjustoff if you are gathering logs across a DaylightSavingsTime (DST) change.

Note The contents of the APPNAME and TAGS fields differ from those previously described in section 5.1.

Table 30: APPNAME and TAGS Used in DUMPLOG Trace Output

Field	Description
APPNAME	PRODUCT_COMPONENT_MESSAGECATEGORY PRODUCT - always ICM COMPONENT – such as Router MESSAGECATEGORY – such as ProcessSynchronization
TAGS	Acceptable tags are: [comp=%s] - component name including side, such as Router A [pname=%s] - process name, such as rtr [iid=%s] - instance name, such as acme [sev=%s] – severity, such as info and optionally [part=%1.%2/%3], which is used only for multi-line entries as described later in this section.

Timestamp

The timestamp displayed in DUMPLOG standard format is in local time relative to the server on which DUMPLOG is run. The timestamp displayed in Cisco Log format is in GMT time independent of the server on which DUMPLOG is run.

Note Date/time options specified on the command line are entered in local time, regardless of whether the Cisco Log option is selected. Therefore, timestamps displayed as part of the Cisco Log formatted entry might appear to be outside of the date/time range selected.

Multi-line Entries

The message portion of some DUMPLOG entries might contain one or more embedded new line characters ('\n'), which cause the messages to appear on multiple lines and might also include blank lines. This is especially true for entries that contain statistics.

For a DUMPLOG standard formatted message, only the first line contains the header field as shown in the following example:

```
00:36:09 ra-nm ICM\acme\RouterA node reporting process statistics for process ccag.
        Process name: ccag
        Process status: A
        Process ID: 6c0
        Number of times process started: 1
        Last start time: 00:35:31 2/8/2007
        Pings completed in zero time: 0
        Pings completed in first third: 0
        Total first third milliseconds: 0
        Pings completed in second third: 0
        Total second third milliseconds: 0
        Pings completed in third third: 0
        Total third third milliseconds: 0
        Longest Ping time: 0
```

For a Cisco Log formatted message, each line contains a separate header. In the example below, however, each entry spans several lines due to page size constraints.

```
19: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.1/14]: ICM\acme\RouterA node reporting process statistics for process
ccag.
20: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.2/14]: Process name: ccag
21: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.3/14]: Process status ACTIVE
22: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.4/14]: Process ID 6c0
23: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.5/14]: Number of times process started 1
24: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.6/14]: Last start time: 00:35:31 2/8/2007
25: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.7/14]: Pings completed in zero time: 0
26: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.8/14]: Pings completed in first third: 0
27: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.9/14]: Total first third milliseconds: 0
28: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.10/14]: Pings completed in second third: 0
29: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.11/14]: Total second third milliseconds: 0
30: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.12/14]: Pings completed in third third: 0
```

```
31: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.13/14]: Total third third milliseconds: 0
32: CICMRGRA: Feb 8 2007 05:36:09.890 +0000: %ICM_Router_unknown:
[comp=Router-A][pname=nm][iid=acme]
[sev=info][part=19.14/14]: Longest Ping Time: 0
```

To differentiate each line in the entry, the part tag is added to each header:

`[part=#1.#2/#3]`

Where:

#1 = the sequence number of the first line (this is the same for all lines in the entry)

#2 = the part number of the specific line

#3 = the total number of parts in the entry

Note the line beginning with sequence number 32, where `[part=19.14/14]`:

#1 = 19. *#2* = 14 / *#3* = 14

Note The log files are zipped according to the parameters specified in the EMS registry settings. While dumping the logs, if one log file transitions to the next log file very quickly, then do one of the following to avoid an error:

- Provide an EndTime (/et) with BeginTime (/bt)
- Increase the file size per log

Note Collecting logs on the UCCE system using dumplog utility impacts CPU and disk utilization. Running dumplog simultaneously on multiple VMs sharing the same disk can cause problems for the disk during peak busy hour if system resources are being stretched near the system limits for BHCA.

For information about system limits for busy hour call attempts (BHCA), see Solution Design Guide for Cisco Unified Contact Center Enterprise at https://www.cisco.com/c/en/us/support/customer-collaboration/unified-contact-center-enterprise/products-implementation-design-guides-list.html.

Some possible workaround include:

- Saving off EMS zip files, and dumping them on an idle system.
- Using System CLI which runs at a lower system priority.
- Serialize dumplog log collection by taking the logs that are likely to wrap first.

EMSMON

While title bar status information is available in the Diagnostic Portico, real time messages can be viewed using EMSMON.

EMSMON displays process messages as they are logged. It displays the same content as the former process windows, except for the title bar and the stdout and stderr output. Logged events for the selected processs appear in the EMSMON window. However, rare error condition messages (for example, shelled processes) that go to stdout do not appear in an EMSMON window.

To change the number of lines each EMSMON window retains, modify the command window parameters.

You can cut and paste in EMSMON (just as in the command windows). It is safer to cut and paste in EMSMON.

For history (events before EMSMON starting), use DUMPLOG.

How to Run EMSMON

You can start EMSMON at anytime, even when the process is not running. (You can have a batch file on a machine to start sessions.) If the process is down, EMSMON displays messages from the process when the process starts. EMSMON does not end when the process ends. To end EMSMON, press **Ctrl+C** or close the window.

EMSMON has the same parameters as ProcMon:

<instance> <node> <process> [*<process>…*] [*<system>*] [*<LanguageID>*]

The system parameter is optional. Use the system parameter to remotely run EMSMON.

For example, if the instance node is “ucce”, to monitor the JTAPI gateway on PG1A, type the following:

EMSMON ucce PG1A jgw1

If you are remote (on another PG) and the system name is UCCEPG1A, type:

EMSMON ucce PG1A jgw1 UCCEPG1A

Note A trust relationship must exist between the two machines. (Use the “NET USE” command or complete an operation that sets up a trust [for example, map a drive].)

The language identification parameter is also optional. As logging is only supported in the English language, it needs to be set to "1033" (for English) whenever the OS is running any other language.

Monitoring Process

Use one EMSMON only for each process.

Run EMSMON Remotely

To reserve system resources for Unified CCE processes, EMSMON can be run from any CCE core system to remotely monitor processes on another CCE system, preferably from an less critical component like a Client Admin Workstation.

EMSMON Connections

You can have one local connection and five remote connections per process. When the number of connections is exceeded, the oldest session is disconnected with the message “You are being disconnected because another user has connected to this named pipe.”

Running EMSMON against a process that is under heavy load is not supported, and can lead to instability in the target process. If your system is running a heavy call load, your EMSMON connections may disconnect and the message “You are being disconnected because the system is running a heavy call load; this connection may impact the performance of the system. Ensure not to reconnect your EMSMON sessions until your system returns to a normal call load.” appear.

Note To prevent Unified CCE processes from exceeding the system memory, Unified CC processes may stop sending queued event messages to slow or paused EMSMON clients. If this occurs, EMSMON clients display a message indicating one of the clients fell behind and there is a gap. This message is also logged in the processes event log. This can happen if a particular EMSMON client is too slow or paused by quick edit or Ctrl+S for example. This does not affect the Unified CCE process, only the EMSMON client.

Unified CCE Certificate Monitoring Service

The Unified CCE Certificate Monitor is a service that monitors the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) based certificates and keys. These certificates and keys are primarily used by the Unified CCE components at the node level. It alerts the system administrator about the validity and expiry of these certificates through Event Viewer. The events from the Unified CCE Certificate Monitoring service are displayed under **Windows Logs > Application**.

This service helps the system administrator to ensure that the systems are installed with valid security certificates without interrupting the Unified CCE services that are running.

Service Installation

Unified CCE Certificate Monitor is installed as a Windows service during ICM installation. Unified CCE Certificate Monitor is not controlled using the Diagnostic Framework Portico. You can view, start, stop, or restart the service from Windows Task Manager.

Certificate Monitoring Events

The Unified CCE Certificate Monitor validates the certificates and keys and reports any error or warning messages to the Event Viewer. The Event Viewer displays the following events:

Table 31: Certificate Monitor Events

Event Type	Event ID	Source	Category
Error	1	CISCO SYSTEMS, INC.ICM	Certificate Monitor
Warning	2	CISCO SYSTEMS, INC.ICM	Certificate Monitor

Certificate and Key Validation

The Unified CCE Certificate Monitor performs the following validations to confirm that a certificate or a key is valid. In case of any discrepancies, the service displays the corresponding error or warning message.

Validation	Description	Error or Warning

Certificate or Key Validation	The certificate and key is validated for:
	• Existence
	• Format Type
	• Integrity
	Note PEM is the only supported certificate format.
		Error:
		Case 1: When the certificate is not available i
		`Certificate <certpath> not found.`
		Case 2: When the key is not available in the d
		`Private key not found.`
		Case 3: When the certificate has an incorrect
		`Failed to load the Security certificat`
		Case 4: When the key has an incorrect format
		`Failed to load the private key. Key fo`
		Case 5: When the key and certificate are not r
		`Certificate <certpath> is not matching`

Subject Validation	The Common Name (CN) is validated for the hostname and domain.	Error: Case 1: When the hostname does not match w `Host name is not matching with Certifi`

Timestamp Validation	Validates the "Not Before" and "Not After" attributes of the timestamp to confirm the certificate or key validation period.	Warning:
		Case 1: When the certificate is not valid befor
		`Certificate <certpath> is not valid be`
		Case 2: When the certificate is not valid after
		`Certificate <certpath> will expire on`

Issuer Validation	Checks if the certificate is a self-signed certificate. A registry value is used to enable or disable this check. By default, the self-signed certificate check is disabled.	Warning: Case: When a self-signed certificate is used: `Self-signed certificate is used.`
Chain Validation	The certificate chain is validated end-to-end. **Note** The root and intermediate CA certificates must be present at the trusted certificates location on the system.	Error: Case: When a certificate in the chain is not fo `Certificate chain validation failed. E`

Serviceability

The Unified CCE Certificate Monitor uses EMS Framework to create and manage its trace files. The certificate monitoring trace files are created in the folder: `<ICM_Drive>:\icm\certmon\logfiles`. You can use the DUMPLOG utility to extract trace files.

Procedure

Step 1 To extract trace files, open command prompt and navigate to the `logfiles` folder.

Step 2 Run the command **C:\icm\certmon\logfiles>dumplog ciscocertmon /<last>/<o>**. In this example, trace files since the last restart are generated.

Supported Log Levels

Unified CCE Certificate Monitor supports four levels of trace configuration based on the level of trace detail and performance impact.

By default, the trace level is set to the level "Error." To change the trace level settings, modify the registry `Software>Cisco System Inc\ICM\CertMon\LogLevel` with the appropriate level.

Trace Level	Description
Error	Has minimal or no performance impact. By default, the log level is set to Error.
Warning	Log more detailed (plus error level) trace messages, small performance impact.
Info	Log more detailed (plus warning or error level) trace messages, medium performance impact.
Debug	Log most detailed (plus error, warning, or info level) trace messages, high performance impact.

Configuration Parameters

For the Unified CCE Certificate Monitor service, the configuration parameters are controlled through the registry at `Software>Cisco System Inc\ICM\CertMon`.

You can configure the following parameters as required:

Configuration Parameter	Description
PollingIntervalInMinutes	The interval in minutes at which the system reports an error or warning.
RejectSelfSignedCertificates	If the value is set to a non-zero value, a warning message is displayed stating that the certificate is a self-signed certificate. This is applicable when the node is configured with a self-signed certificate for its Unified CCE secured operations.
WarningFrequencyInHours	Frequency in hours at which the system displays warning messages. The default value is 24 hours.

Configuration Parameter	Description
WarningThresholdInDays	Frequency in days at which the system displays a warning message for certificate expiry. The default value is 15 days.