Contents

- 1 Purpose
- 2 Deployment Options
  - 2.1 Deployment Option 1: System CLI Configuration with CVP OAMP
    - 2.1.1 Confirm or Add a user to CVP OAMP for the System CLI
    - 2.1.2 Install CVP Remote Operations
    - 2.1.3 Add Remote Operations machines to CVP Operations Console
    - 2.1.4 Confirm Windows Environment Variables are set correctly for CVP Web Services
    - 2.1.5 Start Using the Unified System CLI with CVP OAMP
    - 2.1.6 OPTIONAL: Connecting the CLI to CVP OAMP without Remote Operations
  - 2.2 Deployment Option 2: Devices.csv
    - 2.2.1 Create Devices.csv from Sample file
    - 2.2.2 Populate New devices.csv with connection information
    - 2.2.3 Designate User(s) for Diagnostic Framework
    - 2.2.4 Start Using the Unified System CLI with Devices.csv
    - 2.2.5 Running the System CLI from Multiple Machines with Devices.csv
- 3 Cisco Finesse and System CLI
- 4 Command Examples
  - 4.1 Enter System Mode
  - 4.2 List Connected Devices in System Mode
  - 4.3 Command Modifiers
    - 4.3.1 Setting Tracing on a Device Component
    - 4.3.2 Setting Tracing on a Device Component's Subcomponent
    - 4.3.3 Setting Tracing on a Group
    - 4.3.4 Setting Tracing on a Device Type
    - 4.3.5 On-The-Fly Combination of Devices
  - 4.4 Collecting Logs from Components, Subcomponents, Groups, Device Types
  - 4.5 Collecting Logs & System Snapshot with Show tech-support
  - 4.6 Help with Commands
- 5 Troubleshooting
  - 5.1 Environment Variables
  - 5.2 Unable to Access IOS Devices

Purpose

This guide serves as a simple step-by-step procedure for configuring the Unified System CLI in a customer environment. The goal is to increase awareness of the CLI and push this serviceability tool out to as many deployments as possible. System Mode will make tasks such as setting tracing and grabbing logs significantly easier, and the ?show tech-support? command will be an excellent first step to troubleshooting any problem scenario. The Serviceability Best Practices Guide has complete details on all commands available in the System CLI once configured.

This wiki article also contains some commonly-used command examples after the two deployment options.

Deployment Options

The Unified System CLI is automatically installed on all UCCE components. Out of the box, the CLI will only connect to the local machine. This guide will walk through the minor 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, etc.)
- 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, which is the preferred method for the following reasons:

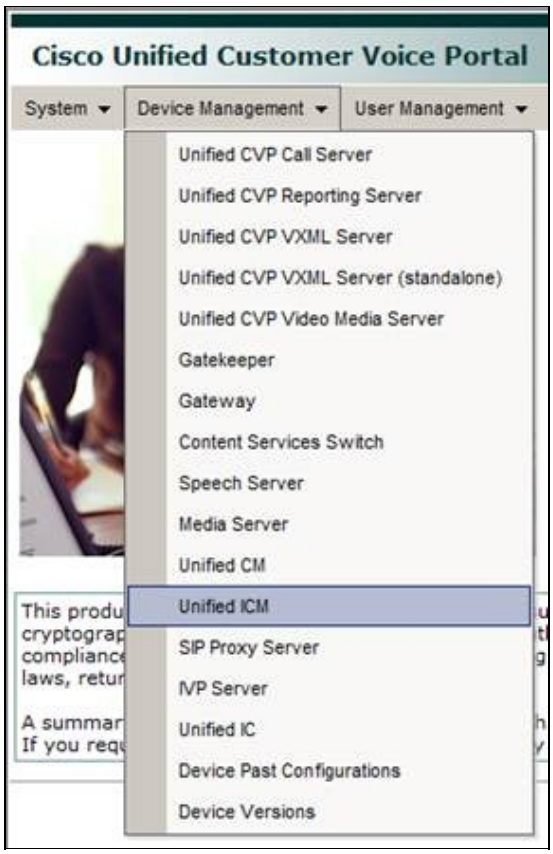
1. All devices are centrally added to and stored in CVP OAMP. One update on OAMP will be reflected in all CLI clients.
2. Passwords for devices are encrypted in OAMP
3. CVP Remote Operations can be installed on any Windows machine, such as a personal laptop, simplifying setup and access to all devices. **Note:** This is not allowed in the CVP 9.0 installer due to CSCub56442 but will be addressed in a future release. In the interim, use the CVP 8.x installer for Remote Operations.

Customers without CVP can still utilize the CLI using a CSV file for connection information. See ?Deployment Option 2: Devices.csv?

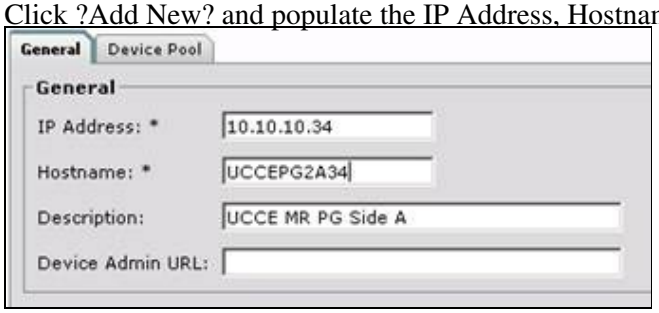
Deployment Option 1: System CLI Configuration with CVP OAMP

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

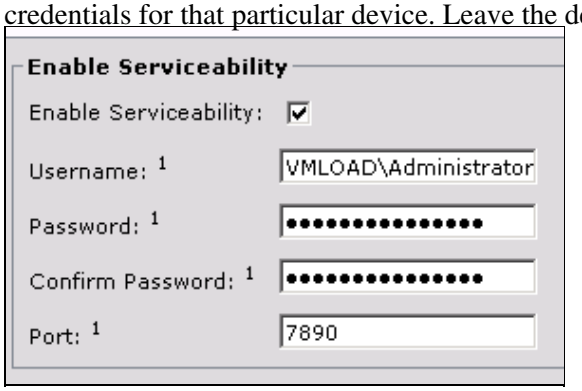
1. Log in to CVP Operations Console from a web browser and navigate to Device Management > Unified ICM.



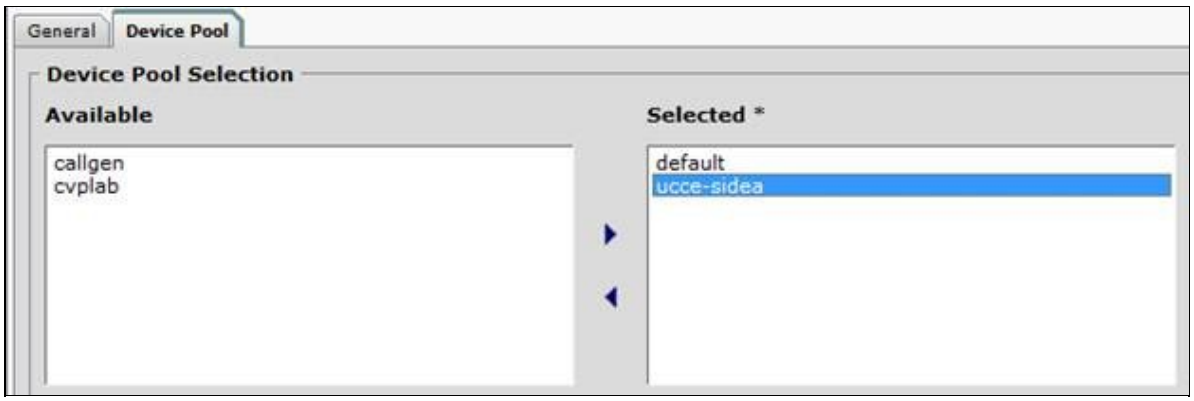
2. Click ?Add New? and populate the IP Address, Hostname, and Description fields.



3. Check the box for Enable Serviceability. Populate the Username and Password fields with login credentials for that particular device. Leave the default port 7890.



4. Click the Device Pool tab and associate the device with a Group if desired. (Tip: Create a ?UCCE-SideA? group for all devices on the A-side!)



5. Click Save.

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

Confirm or Add a user to CVP OAMP for the System CLI

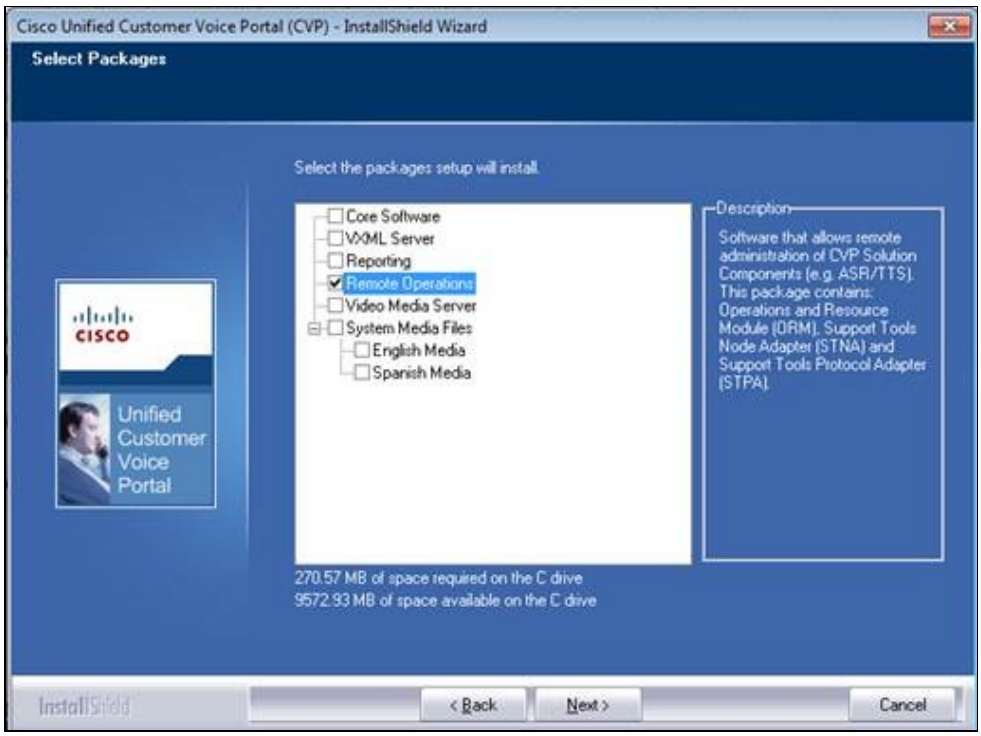
By default on installation, the user ?wsmadmin? is created with the same password as the OAMP Administrator user. If you wish to modify the password for this user, or create a new user, follow these steps.

1. In the CVP Operations Console, click User Management > Users.
2. To modify the existing user, click wsmadmin in the List of Users.
3. To add a new user, click ?Add New.?
4. Once the new username and password have been entered, click the ?User Groups? tab and add the ?ServiceabilityAdministrationUserGroup? to the ?Selected? bucket on the right side.
5. Click Save to complete any updates or additions.

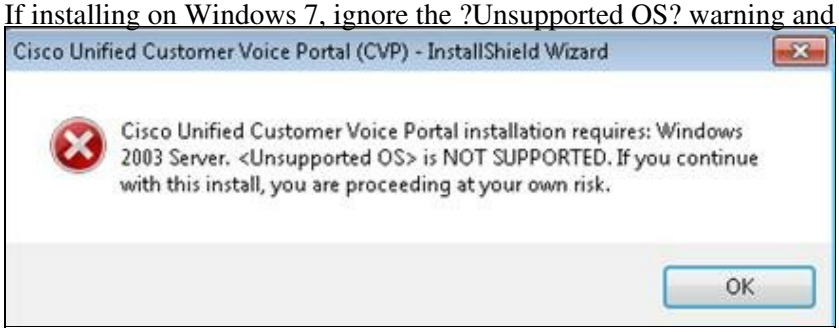
Install CVP Remote Operations

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.

1. Run the CVP 8.5 Installer and select the Remote Operations checkbox.



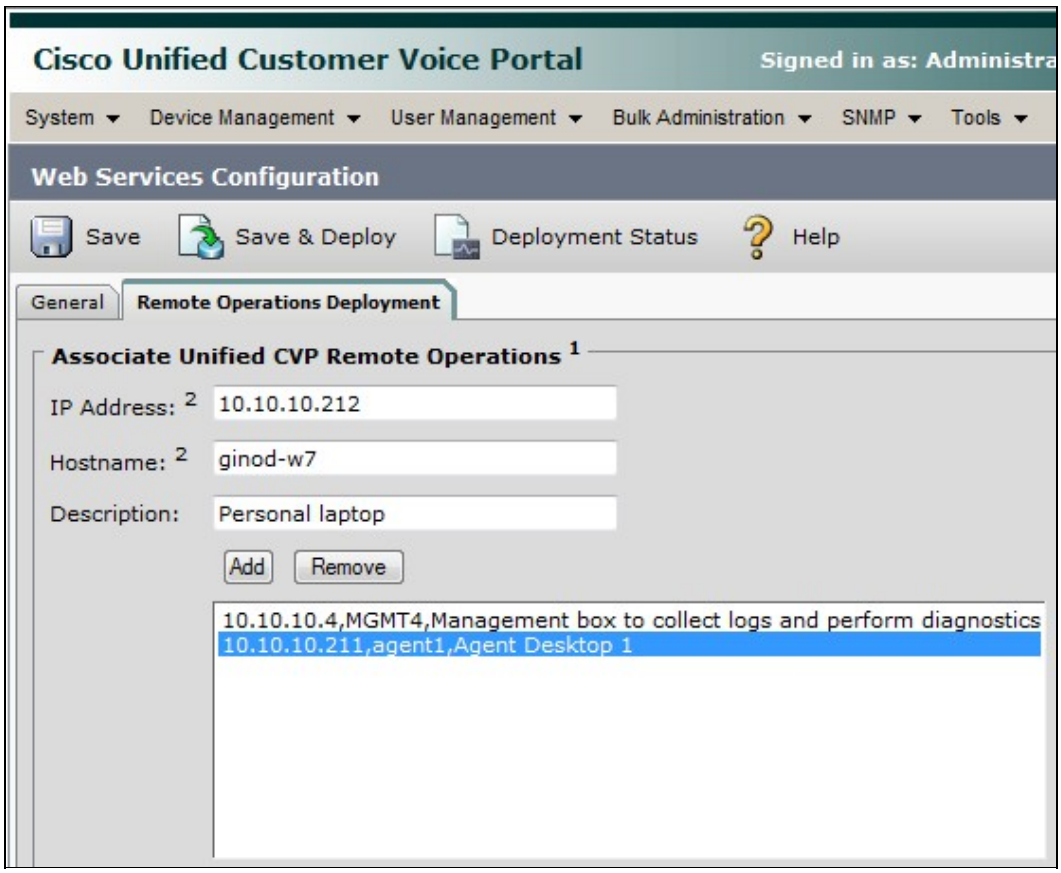
2. If installing on Windows 7, ignore the ?Unsupported OS? warning and click OK.



3. Apply Security Hardening if desired, then complete installation.

Add Remote Operations machines to CVP Operations Console

1. Open a web browser and log in to the CVP Operations Console. Navigate to System > Web Services. Click the ?Remote Operations Deployment? tab on the subsequent screen.
2. Enter the IP Address and Hostname of the machine where CVP Remote Operations was just installed. Include a description if desired and click Add.



3. Repeat for any additional Remote Operations machines, then click ?Save & Deploy? to make this device available for Remote Operations.
4. You will be informed that the Web Services configuration deployment is in progress. Click the ?Deployment Status? button to verify the status of the newly-added machine(s). Click the ?Refresh? button until the status changes to ?Success.?

Confirm Windows Environment Variables are set correctly for CVP Web Services

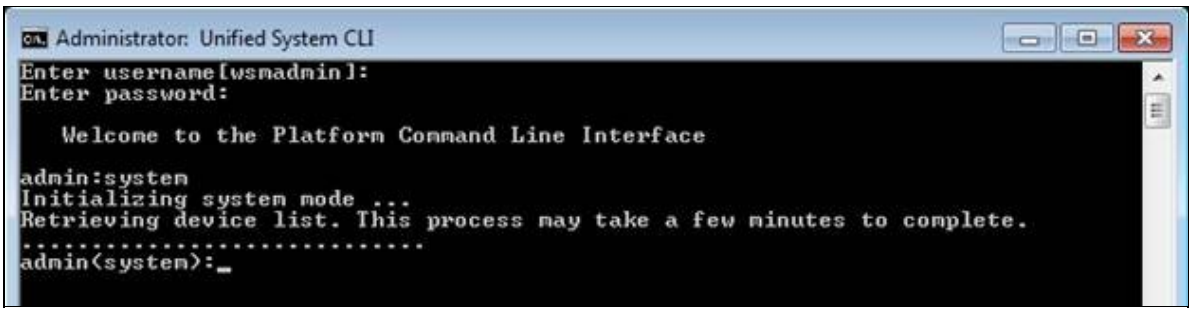
This should have been taken care of by the CVP Remote Operations installation but intermittently fails, so it is important to verify before attempting to connect to the CLI.

1. On the Remote Operations machine, click Start > Run and enter ?systempropertiesadvanced?
2. Click ?Environment Variables?
3. Confirm the System Variable WSC_CLI_DIR is set to ?C:\Cisco\CVP\wsm\CLI?
4. Confirm the Path variable contains ?C:\Cisco\CVP\wsm\CLI;?

Start Using the Unified System CLI with CVP OAMP

Now that the configuration is finished, we?re ready to log in to the CLI and enter System Mode

1. Open the Unified System CLI on the new Remote Operations machine from Start > Programs > Cisco Unified Customer Voice Portal > Unified System CLI.
2. Log in with user wsmadmin (or different if a new user was created).
3. Enter System Mode by typing ?system?. Observe successful servers discovered (indicated by a ?.?) and those that cannot be reached (indicated by ?Unable to connect?).
4. 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.



1. Any changes made in OAMP while a CLI session is active will not be reflected immediately. There are two options for receiving the updates:
    1. Close the console window and start a new connection.
    2. Type ?exit? to leave System Mode and then ?system init?

OPTIONAL: Connecting the CLI to CVP OAMP without Remote Operations

All CCE components automatically include the CLI installed in C:\icm\serviceability\wsccli\. If CVP exists in the deployment but you are for some reason unable to install CVP Remote Operations on its own machine, the pre-installed CLI can be accessed from a CCE machine provided the Environment Variables are configured properly. Open a new Windows command prompt and enter the following:

```
systemcli host: <CVP OAMP host/IP>
```

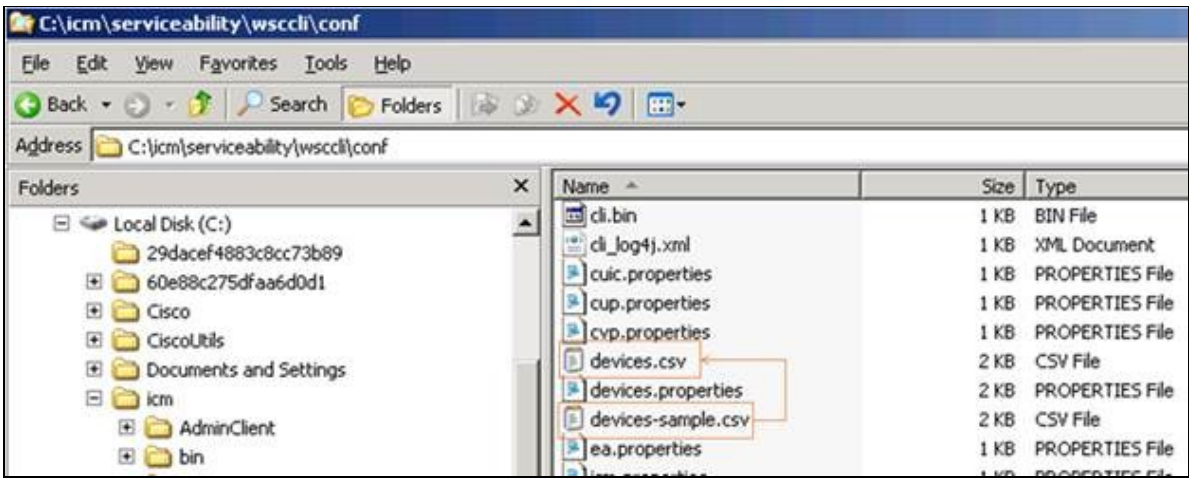
A new System CLI session will be started. Log in with the CVP wsmadmin user (or similar), then enter system mode to download the device list from CVP OAMP.

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

Create Devices.csv from Sample file

Navigate to C:\icm\serviceability\wsccli\conf\. Make a copy of devices-sample.csv and save it as devices.csv.



Populate New devices.csv with connection information

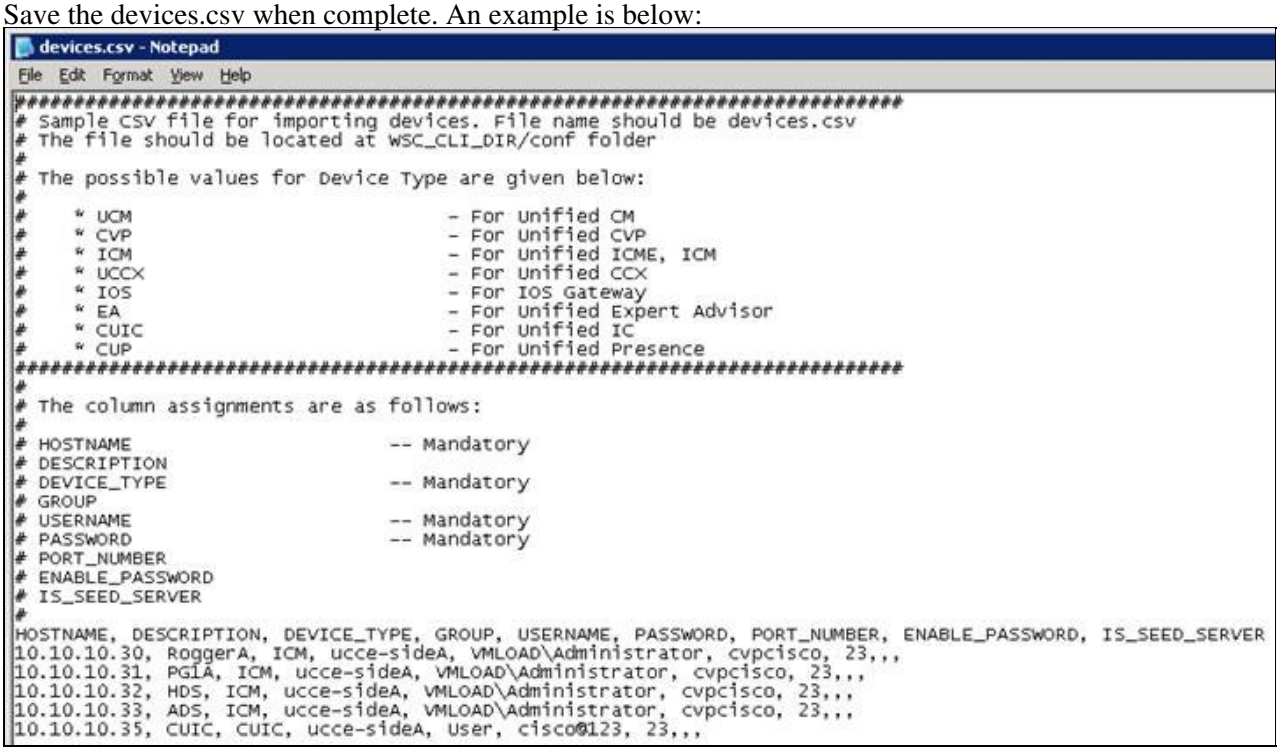
Each device must be added on its own line at the bottom of the devices.csv file. Within each line you must specify the following required fields:

1. IP Address/Hostname
2. Device Type (from the options listed at the top of the file)
3. Username
4. Password ? this is typically the deal-breaker for most customers as it will be stored in plaintext.
5. Port Number (leave the default 23 in most cases)

In addition, the following fields are recommended to make usage easier:

1. Description
2. Group (i.e. UCCE-SideA)

Save the devices.csv when complete. An example is below:

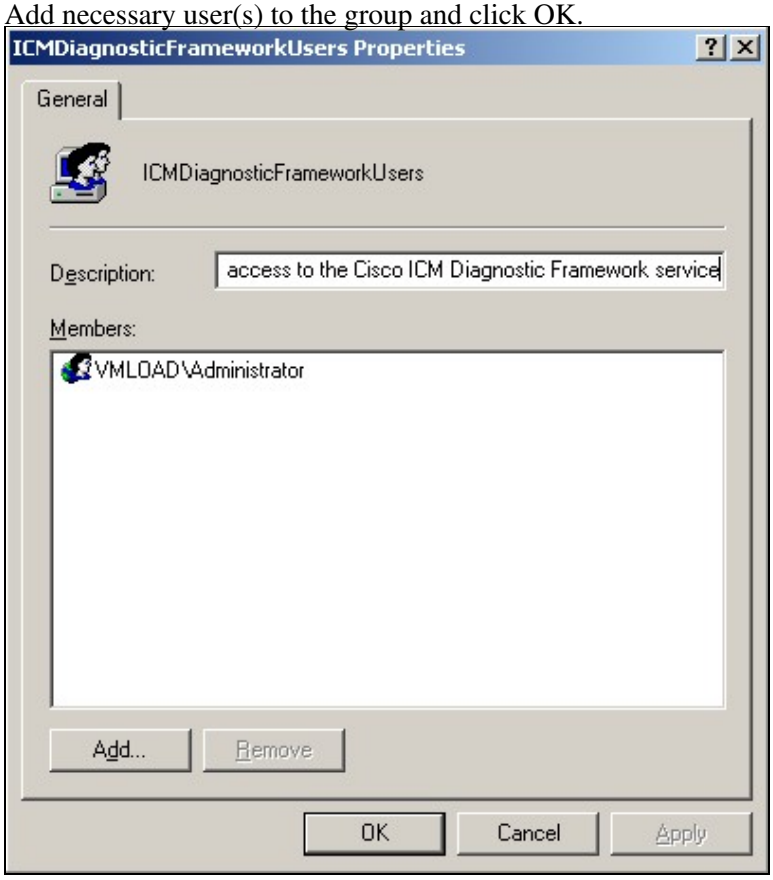


```
devices.csv - Notepad
File Edit Format View Help
##########################################################################
# Sample CSV file for importing devices. File name should be devices.csv
# The file should be located at WSC_CLI_DIR/conf folder
#
# The possible values for Device Type are 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_NUMBER, ENABLE_PASSWORD, IS_SEED_SERVER
10.10.10.30, RoggerA, ICM, ucce-sideA, VMLOAD\Administrator, cvpcisco, 23,,,
10.10.10.31, PGIA, 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 User(s) for Diagnostic Framework

Users must be a part of the Local Group ?ICMDiagnosticFrameworkUsers? in order to initially log in to the CLI when using devices.csv. To validate and add users to this group:

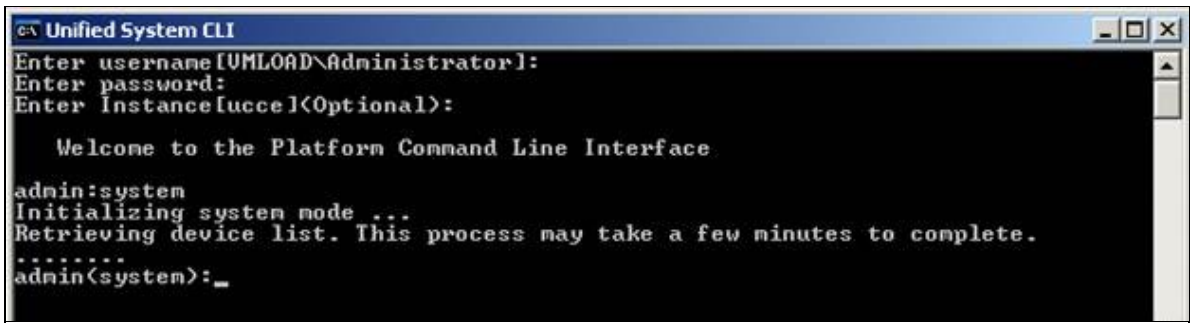
1. Click Start > Run and enter ?lusrmgr.msc?
2. Click the Groups folder and double-click ?ICMDiagnosticFrameworkUsers?
3. Add necessary user(s) to the group and click OK.



Start Using the Unified System CLI with Devices.csv

On the ADS, click Start > Programs > Cisco Unified CCE Tools > Unified System CLI. (If this shortcut is missing for some reason, run C:\icm\serviceability\wsccli\runwsccli.bat)

1. Log in with a member of the ICMDiagnosticFrameworkUsers group
2. 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.
3. Once successfully logged into the local machine, enter System Mode by typing ?system?. Observe successful servers discovered (indicated by a ?.?) and those that cannot be reached (indicated by ?Unable to connect?).
4. 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



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 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.

Cisco Finesse and System CLI

Starting with Finesse Release 10.0(1), you can use System CLI to collect Finesse logs.

Starting with Finesse Release 10.5(1), you can use System CLI to get performance (show perf) and version (show version) information using REST support.

If Finesse is deployed with Unified CCE Release 9.x or Release 10.0(x), you must perform the following configuration on the Unified CCE ADS machines.

Note: This procedure is required if Finesse is deployed with Unified CCE Release 9.x or Release 10.0(x) only. If Finesse is deployed with Unified CCE Release 10.5(1), skip this procedure.

1. In the **C:\icm\serviceability\wsccli\conf** directory, open the **devices.properties** file.
2. Add the new device type **finesse**.
   The device type line should read as follows:

   ```
   Devices=icm cvp ucm ios ea uccx cuic cup finesse ccdm ccmp rsm sm
   ```
3. In the **C:\icm\serviceability\wsccli\conf** directory, create a new file called **finesse.properties** (or rename the **fin.properties** file).
4. Put content in the **finesse.properties** file as shown in the following examples for your release:

   finesse.properties file for Finesse Release 10.5(1):

   ```
   # Protocol supported; rest, soap, telnet
   # by default is rest and apply to all commands; each device type can override it if needed
   # Each protocol must have a port defined for the connection
   Protocol=Rest Soap

   Rest.Port=8443
   Rest.Commands=perf version

   Soap.Port=8443
   Soap.Commands=debug debug_level license platform trace version
   ```

   finesse.properties file for Finesse Release 10.0(1):

   ```
   #Protocol supported; rest, soap, telnet
   # by default is rest and apply to all commands; each device type can override it if needed
   ```

```
# Each protocol must have a port defined for the connection
Protocol=Soap

Sopa.Port=8443
Soap.Commands=debug debug_level license platform trace version
```

Note: Finesse does not support System CLI to set system trace setting.

Command Examples

Now that the System CLI is configured, let's run some commands! There is an excellent Support Forums post on using the CLI for individual servers. This wiki article will expand on that post with System Mode commands.

Enter System Mode

`admin:` **system**

```
Initializing system mode ...
Retrieving device list. This process may take a few minutes to complete.
.....
```

List Connected Devices in System Mode

`admin(system):` **show devices**

```
Device host name: 10.10.10.30
Description: RoggerA
Port: 23
Password: ********
Alternate user:
Enable password: ********
Product type: ICM
Group: ucce-sideA

Device host name: 10.10.10.31
Description: PG1A
Port: 23
Password: ********
Alternate user:
Enable password: ********
Product type: ICM
Group: ucce-sideA

Device host name: 10.10.10.33
Password: ********
Enable password: ********
Product type: ICM
Product component type: Cisco ICM Diagnostic Framework
```

Command Modifiers

Once in System Mode, commands can be run against a targeted list of devices. Below are examples of setting debug level 1 with various subsets of devices. Note that these modifiers can be used for commands other than setting tracing!

Setting Tracing on a Device Component

A "component" is the equivalent of an entire service - i.e. Router A, Logger A, PG1A, etc.

`admin(system)` **: debug level 1 server 10.10.10.30 component "icm:Peripheral Gateway 1A"**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [debug_level] data from device [10.10.10.30] ProductType [icm] ...

Component: icm:Peripheral Gateway 1A, subcomponent: icm:jgw2
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:opc
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pgag
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim1
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim2
Debug level is set successfully
```

Setting Tracing on a Device Component's Subcomponent

A subcomponent is an individual process within the service - i.e. PIM1 or OPC within PG1A.

`admin(system)` **: debug level 1 server 10.10.10.30 component "icm:Peripheral Gateway 1A" subcomponent icm:pim1**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [debug_level] data from device [10.10.10.30] ProductType [icm] ...

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim1
```

```
Debug level is set successfully
```

Setting Tracing on a Group

Group membership is defined with each device's connection information in CVP OAMP->Device Pool or devices.csv. Groups can be created with any combination of devices and device types - IOS, ICM, CVP, UCM, etc.

`admin(system)` **: debug level 1 group ucce-sideA**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [debug_level] data from device [10.10.10.30] ProductType [icm] ...

Component: icm:CTI Server 1A, subcomponent: icm:ctisvr
Debug level is set successfully

Component: icm:CTI Server 1A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:CTI Server 1A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:ctios
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nm
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:cms
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:cmsjserver
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:ise
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rtc
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rtd
```

```
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:uaw
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:baImport
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:CampaignManager
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:csfs
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:cw2kFeed
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:dtp
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:hlgr
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:rcv
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:jgw2
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:opc
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pgag
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim1
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim2
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:agi
Debug level is set successfully
```

```
Component: icm:Router A, subcomponent: icm:ccag
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:dba
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:dbw
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nms
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:rtr
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:rts
Debug level is set successfully

Component: icm:Cisco ICM Diagnostic Framework
Debug level is set successfully

Component: icm:Web Setup
Debug level is set successfully
```

Setting Tracing on a Device Type

A device type command will apply to all connected devices of that type - all ICM, or all CVP, or all UCM, etc.

`admin(system)` **: debug level 1 devicetype icm**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [debug_level] data from device [10.10.10.30] ProductType [icm] ...

Component: icm:CTI Server 1A, subcomponent: icm:ctisvr
Debug level is set successfully

Component: icm:CTI Server 1A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:CTI Server 1A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:ctios
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nm
```

```
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:cms
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:cmsjserver
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:ise
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rtc
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:rtd
Debug level is set successfully

Component: icm:Administration and Data Server, subcomponent: icm:uaw
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:baImport
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:CampaignManager
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:csfs
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:cw2kFeed
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:dtp
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:hlgr
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Logger A, subcomponent: icm:rcv
Debug level is set successfully
```

```
Component: icm:Logger A, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:jgw2
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:opc
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pgag
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim1
Debug level is set successfully

Component: icm:Peripheral Gateway 1A, subcomponent: icm:pim2
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:agi
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:ccag
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:dba
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:dbw
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:nms
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:rtr
Debug level is set successfully

Component: icm:Router A, subcomponent: icm:rts
Debug level is set successfully

Component: icm:Cisco ICM Diagnostic Framework
Debug level is set successfully

Component: icm:Web Setup
Debug level is set successfully
```

```
Retrieving [debug_level] data from device [10.10.10.130] ProductType [icm] ...

Component: icm:CTI Server 1B, subcomponent: icm:ctisvr
Debug level is set successfully

Component: icm:CTI Server 1B, subcomponent: icm:nm
Debug level is set successfully

Component: icm:CTI Server 1B, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:ctios
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nm
Debug level is set successfully

Component: icm:CTI OS Server 1, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:cms
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:cmsjserver
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:ise
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:rtc
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:rtd
Debug level is set successfully

Component: icm:Bdministration and Data Server, subcomponent: icm:uaw
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:baImport
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:CampaignManager
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:clgr
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:csfs
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:cw2kFeed
```

```
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:dtp
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:hlgr
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:rcv
Debug level is set successfully

Component: icm:Logger B, subcomponent: icm:rpl
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:jgw2
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:nmm
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:opc
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:pgag
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:pim1
Debug level is set successfully

Component: icm:Peripheral Gateway 1B, subcomponent: icm:pim2
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:agi
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:ccag
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:dba
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:dbw
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:mds
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:nm
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:nmm
Debug level is set successfully
```

```
Component: icm:Router B, subcomponent: icm:nms
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:rtr
Debug level is set successfully

Component: icm:Router B, subcomponent: icm:rts
Debug level is set successfully

Component: icm:Cisco ICM Diagnostic Framework
Debug level is set successfully

Component: icm:Web Setup
Debug level is set successfully
```

On-The-Fly Combination of Devices

Some commands can also be run on multiple devices without a common group or device type by using the pipe "|" character.

`admin(system)` **: show config server 10.10.10.30|10.10.10.31 redirect dir c:\temp**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [configuration] data from device [10.10.10.30] ProductType [icm] ?
Fetching configuration resource for component DumpCfg
Fetching configuration resource for component ExportICMCfg
Fetching configuration resource for component ConfigExport
Fetching configuration resource for component Registry

Retrieving [configuration] data from device [10.10.10.31] ProductType [icm] ?
Fetching configuration resource for component Registry

Output is saved to ?c:\temp\clioutput0.zip?
```

Collecting Logs from Components, Subcomponents, Groups, Device Types

Logs can then be collected using the same parameters from above. In addition, since you are collecting a file, you must tell the CLI where to save it with the "redirect dir" parameter.

`admin(system)` **: show trace server 10.10.10.30 component "icm:Router A" redirect dir c:\temp**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [trace] data from device [10.10.10.30] ProductType [icm] ...

Default time range is last 24 hours.
Downloading file: [RouterA[ucce]_agi_20120601161156149_5373904.zip], date: [Thu May 31 16:1
```

```
Downloading file: [RouterA[ucce]_ccag_20120601161157648_1753611.zip], date: [Thu May 31 16:

Downloading file: [RouterA[ucce]_dba_20120601161157998_8896691.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_dbw_20120601161158341_6039772.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_mds_20120601161158682_3182853.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_nm_20120601161159037_8689731.zip], date: [Thu May 31 16:11

Downloading file: [RouterA[ucce]_nmm_20120601161159401_4196609.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_nms_20120601161159744_1339690.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_rtr_20120601161200096_2070821.zip], date: [Thu May 31 16:1

Downloading file: [RouterA[ucce]_rts_20120601161200460_7577699.zip], date: [Thu May 31 16:1

Output is saved to "c:\temp\clioutput4.zip"
```

Collecting Logs & System Snapshot with Show tech-support

By default, the show tech-support command collects a 24-hour snapshot of ALL defined server component logs (including OPC Capture), ICM config, dumpcfg, versions, registries, OS info, PerfMon, debug levels, etc. To adjust this command to capture more or less than 24 hours, use the "absdatetime" parameter.

`admin(system)` **: show tech-support absdatetime 5-31-2012:09:00 5-31-2012:12:00**

```
Warning: Because running this command can affect system performance,
Cisco recommends that you run the command during off-peak hours.
Do you want to continue? [y/n]: y

Retrieving [version] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [version] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [version] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [component] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [component] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [component] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [log] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [log] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [log] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [trace] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [trace] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [trace] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [configuration] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [configuration] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [configuration] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [debug] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [debug] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [debug] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [license] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [license] data from device [10.10.10.31] ProductType [icm] ...
```

```
Retrieving [license] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [perf] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [perf] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [perf] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [platform] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [platform] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [platform] data from device [10.10.10.33] ProductType [icm] ...

Retrieving [sessions] data from device [10.10.10.30] ProductType [icm] ...
Retrieving [sessions] data from device [10.10.10.31] ProductType [icm] ...
Retrieving [sessions] data from device [10.10.10.33] ProductType [icm] ...

Output is saved to "C:\icm\serviceability\wsccli\.\download\clioutput1.zip"
```

Help with Commands

Available parameters and additional commands can be found using the "?" character within the CLI, just like in IOS.

`admin(system)` **: show debug server 10.10.10.30 component ?**

```
Options: cvp:CallServer
         cvp:OAMP
         cvp:ORM
         cvp:VXMLServer
         icm:Administration and Data Server
         icm:CTI OS Server 1
         icm:CTI Server 1A
         icm:Cisco ICM Diagnostic Framework
         icm:Logger A
         icm:Peripheral Gateway 1A
         icm:Peripheral Gateway 2A
         icm:Router A
         icm:Web Setup
```

The Serviceability Best Practices Guide also has complete details on all commands available in the System CLI once configured.

Troubleshooting

There are a few areas where issues are commonly encountered.

Environment Variables

Depending on the Deployment Type used, the WSC_CLI_DIR and PATH variables **on the machine running the CLI** (i.e. personal laptop, ADS) must contain specific directories.

1. Click Start, type systempropertiesadvanced and hit Enter.
2. Click Environment Variables
3. Check the system variable WSC_CLI_DIR

    1. If using CVP Remote Operations, this should be set to C:\Cisco\CVP\wsm\CLI
    2. If using devices.csv, this should be set to C:\icm\serviceability\wsccli
4. Check the Path variable
    1. If using CVP Remote Operations, this should be set to C:\Cisco\CVP\wsm\CLI
    2. If using devices.csv, C:\icm\serviceability\wsccli; should be included in the full string of the Path variable. Ensure C:\Cisco\CVP\wsm\CLI; is NOT present.

Unable to Access IOS Devices

Note that OAMP requires telnet access to the voice gateways in order for the IOS-specific System CLI commands to work. If telnet access to network devices is restricted, the CLI may not work properly for IOS devices.