Cisco IMC Plugin Release 0.9.4 - User Guide

For Nagios Core

February 28, 2017

Table of Content

List of Tables

List of Figures

1. Overview

Data center administrators have been using Nagios for more than a decade now and it has emerged as one of the favorite open source tool for the Data Center monitoring.

Nagios is an open source computer system monitoring, network monitoring and infrastructure monitoring software application. Nagios offers monitoring and alerting services for servers, switches, applications, and services.

The solution provides end-user with two primary components.

The first is the Nagios monitoring plugin script which will provide end-user with the capability of monitoring the Cisco IMC of standalone UCS C-series servers.

The second is an add-on to the Nagios, which will provide end-user with the capability to auto discover Cisco IMC of standalone UCS C-series servers.

1.1 Acronyms and Abbreviations

The following table describes the acronyms and abbreviations used in the document.

Abbreviation	Translation
DNS	Domain Name Server
FQDN	Fully Qualified Domain Name
IMC	Integrated Management controller
NAGIOS_HOME	Nagios install directory.
NAGIOS_ETC_DIR	Nagios etc directory path
NAGIOS_PLUGIN_DIR	Nagios plugin directory path
NAGIOS_LOGOS_DIR	Nagios logos image directory

1.2 System Requirements

The Nagios must meet the below mentioned minimum requirements for this solution to work

- Operating System – Nagios supported Linux server.
- http://www.nagios.org/about/propaganda/distros
- Nagios Core 3.2.x or higher
- http://www.nagios.org/download/core/thanks/?t=1398749242
- Latest Nagios Plugins
- http://www.nagios.org/download/plugins/
- Latest Cisco IMC Python SDK 0.9.2 or higher. It will not work with IMC Python SDK < 0.9.2.
- pip install imcsdk

1.3 What's new

- Version 0.9.4
  - Support for modular C3260 and classic platforms
  - Requires the newer imcsdk 0.9.2 and above
  - Supports IMC firmware version 1.5 and above
  - Support for TLSv1.2 on IMC firmware version 3.0 and above
  - Issue fixed for fetching all faults on the server when querying by 'faultInst' Managed Object's class-id or dn
  - Hierarchy "-R flag" is ignored when querying by 'faultInst' Managed Object's class-id or dn
  - Issue fixed for IP range support
  - Fixed documentation

2. Installing the solution

The solution is provided in the tar gzip format which can be easily extracted in any of the Nagios supported Linux distributions.

The tar gzip file extracts to a folder named cisco-imc-nagios-x.x.x (x.x.x here is the build version) and contains following three files

- `INSTALL` – This file guides end user on how to install the solution in an existing Nagios installation
- cisco-imc-nagios-x.x.x`.tar.gz` – This tar gzip contains the Cisco IMC monitoring plugin and autodiscovery add-on tar gzip.
- `installer.py` – This is installer script which uses the cisco-imc-nagios-x.x.x`.tar.gz` tar gzip and installs the solution as per the user environment.

2.1 Install Paths

Installation requires that you are aware of the paths to the following locations and they will depend on your Nagios installation as your environment. Please check with your Nagios administrator for more information.

Listed below are typical install locations and directories for different linux distributions

- For Debian/SUSE
    - The Nagios home directory
      `NAGIOS_HOME=/etc/nagios3`
    - Nagios etc directory that has nagios configuration files
      `NAGIOS_ETC_DIR=/etc/nagios3`
    - Nagios plugin directory that has all the Nagios plugin
      `NAGIOS_PLUGIN_DIR=/usr/lib/nagios/plugins`
    - Nagios logos directory
      The logos directory is generally a part of CGI configuration file and the root path to the logos directory is denoted by ‘`physical_html_path`'.

      Appending `'images/logos'` to the value of the above variable provides us the logos directory path for Nagios.
      The `cgi.cfg` file can be found in `NAGIOS_ETC_DIR`

      `NAGIOS_LOGOS_DIR=/usr/share/nagios3/htdocs/images/logos/`
- In other Linux variants, typical paths can be

```
NAGIOS_HOME=/usr/local/nagios
NAGIOS_ETC_DIR=/usr/local/nagios/etc
NAGIOS_PLUGIN_DIR=/usr/local/nagios/libexec
NAGIOS_LOGOS_DIR=/usr/local/nagios/share/images/logos/
```

2.2 Install Nagios Monitoring Plugin and Auto Discovery

Following are the step for installing Cisco IMC integrations for Nagios.

a. Extract the installation tar gzip file in a temporary location.

```
# tar zxvf cisco-imc-nagios-x.x.x.tar.gz
```

b. Now run the installer, which should be present in the extracted folder. Installer by default will install the monitoring plugin along with the auto discovery add-on scripts. It also auto detect various install paths and prompt with default options for installing this plugin.

```
# ./installer.py
```

c. Installer also updates the configuration files which are required for the working of this plugin. It prompts and creates the backup of all the files which will be modified in this process.

d. In case, only monitoring plugin is to be installed then use the '`--plugin`' option

```
# ./installer.py --plugin
```

3. Using Auto Discovery Nagios

In the autodiscovery directory, add/update IMCHostInfo.csv with the Cisco IMC IP/FQDN and login credentials. User can also use the CLI parameters if a single domain or a range of IPs needs to be discovered.

Below is the parameter mapping of CSV file to CLI parameters:

CSV Parameter	CLI Parameter
HostName	-H, --host
User	-u, --user
Password	-p, --password
Port	-n, --port
NoSSL(True/False)	--NoSsl
Proxy URL	--proxy

Table 1 : CSV to CLI parameter mapping

The servers that are defined in this CSV/CLI will be discovered and added to Nagios for monitoring.

Example CSV:

```
<HostName>,<Username>,<Password>,<Port>,<NoSSL(True/False)>,<ProxyURL>
192.168.1.10,admin,password,80,True
192.168.1.10,admin,password,80,True,http://proxy.ip.com:8080
192.168.1.10-15,admin,password
```

Here the HostName, User and Password fields are mandatory for the auto discovery plugin to discover the Cisco IMC servers.

User can provide IP range in the hostname. Auto-Discovery script allows range definition by passing "-" in the fourth octet. For all IPs in that range, connection parameters will be same i.e. the username, password, port, SSL and proxy data if applicable.

As given in the above CSV entries, there is an entry "192.168.1.10-15".
This range will be expanded by Auto-discovery script as shown below:

```
192.168.1.10, admin, password
192.168.1.11, admin, password
...
...
192.168.1.15, admin, password
```

Note:

a) In case user password contains any special character then it has to be provided in double quotes.

```
192.168.1.16,admin,"pass,word"
192.168.1.16,admin,"My_password"
```

b) In case user password contains "(double quotes) then it has to be escaped by another "(double quotes). If password is my"password then we will write the same in csv file as given below.

```
192.168.1.16,admin,"my""password"
```

c) In case user is a domain user then the user field should be defined as "<User>@<Domain>"

```
192.168.1.16,"admin@somedomain.cisco.com",abcd12345
```

d) Giving IP range is allowed only for IPv4 addresses.

Now run the auto discovery script

- If using CSV file:

  #./NagiosAutoDiscoveryIMC.py

- If using CLI parameters:

  #./NagiosAutoDiscoveryIMC.py –H <Host Name> -u <user name> -p <password>

Note:

Input via CLI will be given preference over CSV file, i.e. if the Host Parameter via CLI is given then script will skip reading the CSV file.

3.1 Default Auto Discovery Config

```
==========================================================================
# This configuration file is used as a placeholder
# for externalizing variables which may differ
# from one nagios deployment to other

# User should update the following variables as per
# his Nagios deployment
NAGIOS_HOME=/usr/local/nagios/etc
NAGIOS_RESTART_CMD=/etc/init.d/nagios restart

# User can provide his own service name and service base type here.
# The list below should be in the following format
# <service name>:<class or dn> ,<service name>:<class or dn>:<optional cli options>
# here class or dn can be provided for creating the service.
# Example :
# SERVICE_LIST= Check Rack Server:ComputeRackUnit:"--inHierarchical --faultDetails --
useSharedSession",Check CPU:ProcessorUnit:"--useSharedSession"
#SERVICE_LIST= Check Rack Server:ComputeRackUnit:"--inHierarchical --faultDetails --
useSharedSession",  Monitor Faults:FaultInst:"-R --useSharedSession"
#SERVICE_LIST= Check Rack Server:EquipmentChassis:"--inHierarchical --faultDetails --
useSharedSession",  Monitor Faults:FaultInst:"-R --useSharedSession"
SERVICE_LIST= Monitor Server Health:FaultInst:"-R --useSharedSession"


#If this flag is set to False, then we will not remove the previously discoverd objects.
#If this flag is set to True , then the old discoveries will be removed. This is the default behaviour.
REMOVE_OLD_DISCOVERY=TRUE
#REMOVE_OLD_DISCOVERY=FALSE

## For Developer Use ##
host_group_xml_file = "<NAGIOS_HOME>/cisco/imcObjs/.hostgrp.xml"
host_group_file = "<NAGIOS_HOME>/cisco/imcObjs/hostgrp.cfg"
service_xml_file = "<NAGIOS_HOME>/cisco/imcObjs/<IMC_IP>/.services.xml"
service_cfg_file = "<NAGIOS_HOME>/cisco/imcObjs/<IMC_IP>/services.cfg"
host_defination_file = "<NAGIOS_HOME>/cisco/imcObjs/cisco_imc_host.cfg"
cmd_conf_file = "<NAGIOS_HOME>/cisco/imcObjs/cmd.cfg"
host_ext_info_xml_file = "<NAGIOS_HOME>/cisco/imcObjs/.hostextinfo.xml"
host_ext_info_file = "<NAGIOS_HOME>/cisco/imcObjs/hostextinfo.cfg"
==========================================================================
```

4. Features

Once the installation is complete and auto discovery script is executed, user can now see the Cisco IMC servers which are discovered.

4.1 Maps View

The discovered IMC will be displayed in the Map section as shown below.

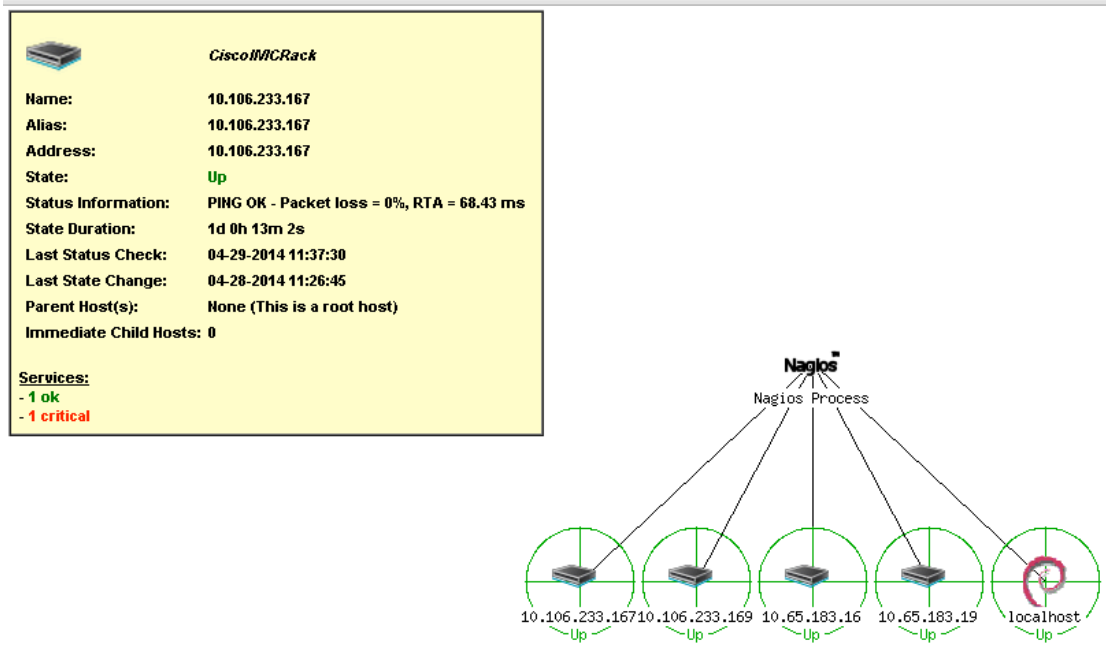

Figure 1 : Cisco IMC Fault in Nagios

4.2 Service View

The monitoring plugin for IMC not only monitors the health status of the IMC devices but also provides the relevant inventory information in case no fault has occurred on the given component.

The default monitoring service for the rack server checks for all the faults that may have occurred for the given server.

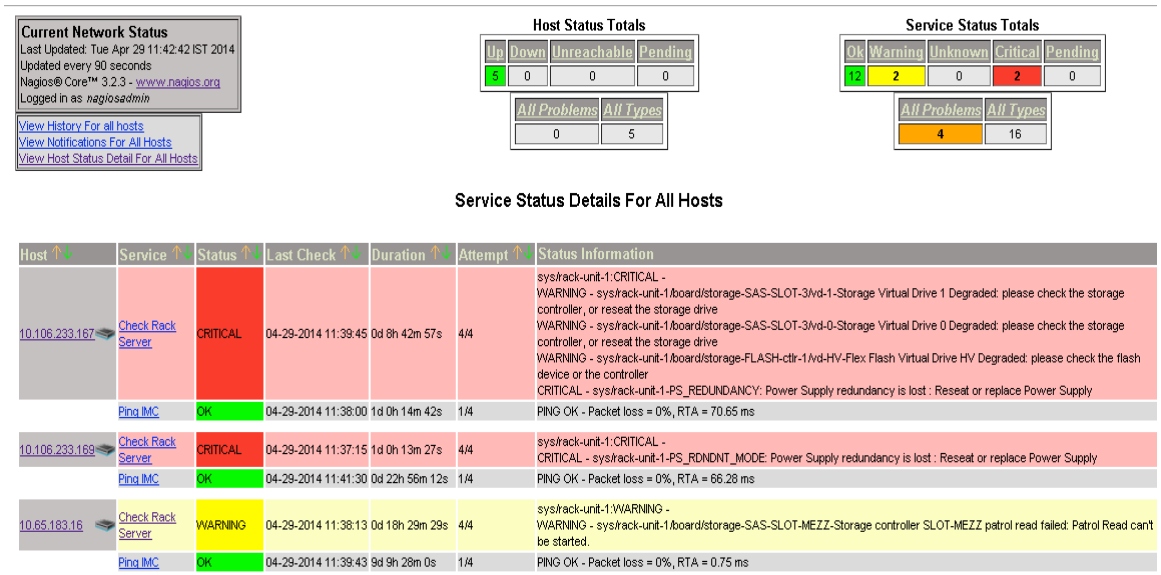

Figure 2 : Cisco IMC Service View in Nagios

Based on the IMC fault information the plugin decides the Nagios service state.

The following table shows the mapping of the severity levels of the Cisco IMC faults to Nagios States.

IMC Fault Severity	Nagios States
Critical and Major	CRITICAL
Minor and Warning	WARNING
Info and Cleared	OK

Table 2 : Cisco IMC Fault Severity to Nagios State Mapping

4.3 Detail Fault View

The monitoring plugin for Cisco IMC will fetch the relevant faults details for a given dn or class.

For example, a fault as major on the server will be depicted as critical and the fault details will be shown on the service state information page of the Nagios service.

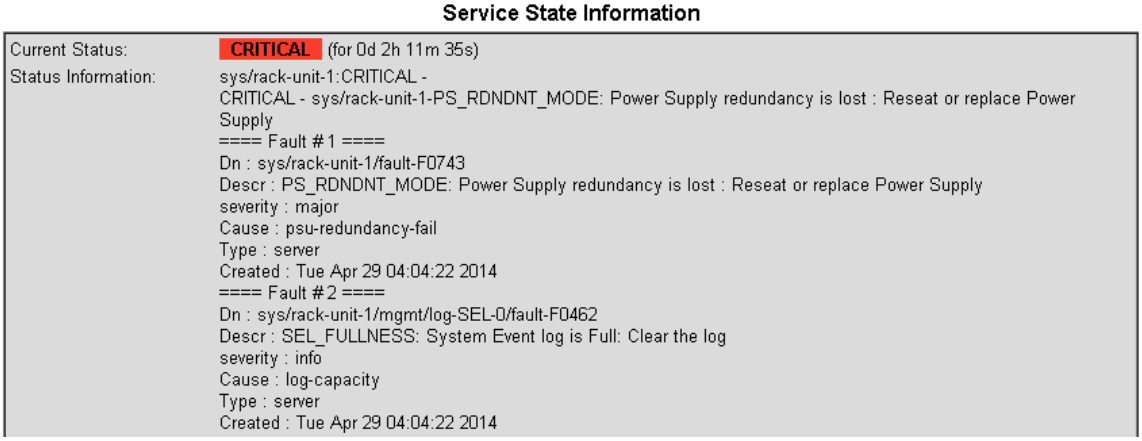

Figure 3 : Cisco IMC Fault Details View in Nagios

5. Monitoring Plugin

5.1 Plugin Script

As per the Nagios standards, the Cisco IMC Nagios monitoring plugin takes multiple standard inputs like the host information, connection information and service status criteria. The plugin is named as "*cisco_imc_nagios*" and can take the following cli inputs

Argument Name	Description	Remarks
`-H/ --host`	IP/FQDN of an IMC host	Required
`-n/--port`	CIMC http/https port	Optional
`--NoSsl`	If defined this flag will turn off the secure communication (SSL) with the IMC host	Optional
`-u/--user`	CIMC login user name. **Note:** In case user is a domain user then the user field should be defined as "<User>@<Domain>" Example " admin@somedomain.cisco.com"	Required
`-p/--password`	CIMC login password	Required, if --passEnc is not provided.Either of the one should be provided.
`--passEnc`	Base64 encoded password for internal framework communication. Services added by auto discovery uses this variable.	Required, if -p or --password is not provided. Either of the one should be provided.
`-t/-type`	Query type, i.e class or dn. Example `-t class` or `--type dn`.	Required
`-q/--query`	Value of query class or dn. Example if -t is dn then -q should be a dn like "`–q sys/rack-unit-1`" or "`–q sys/chassis-1/server-1`" If -t is class then -q should be "`-q ComputeRackUnit`" or "`–q ComputeServerNode`"	Required
`-a/ --attribute`	Attribute that user wants to fetch value for. User can either provide just the attribute name or user can provide the attribute name and the user given name which will be displayed on the Nagios web UI. So, user can either provide just the -a <attribute name> or --attribute <attribute name>:<user given name> If <user given name> is not provided then the <attribute name> will be displayed in the Nagios web UI along with its value. If <user given name> is provided the	In addition, user can provide –r or -w and –c parameters

	web UI will display the <user given name> along with the attribute value.	
-w/ --warn	User defined Warning level	In case user provides –a or –c then this is required else optional
-c/ --critical	User defined Critical level	In case user provides –a or –w then this is required else optional.
-r/ --regex	Regular Expression for matching the pass condition which will result in service status in Nagios as 'OK' else it will be 'CRITICAL'	In case user provides –a, –c or –w then this is required else optional.
--proxy	Proxy URL for connecting to the IMC servers. Example --proxy "http://<Proxy IP>:<Port>"	Optional
-R/ --inHierarchical	If specified this will provide an overall hierarchical health status for all the elements under the given class or dn.	Optional
-F/ --faultDetails	If specified it will work with inHierarchical flag and will look for fault details under the given class or dn. It is quick way for checking the overall status of the given dn or class	Optional
-b/--blackListRegexp	If specified this will exclude the dn which are defined by this regular expression.	Optional
--debug	If defined it will print the xml communication between the plugin and IMC Host. It is also helpful for detailed debugging in case of any error that may have occurred while using this CLI. Use this for debug purpose only.	Optional
-S/--useSharedSession	If specified it will try and reuse an existing IMC connection for a specified user. If the connection does not exist, then a session will be created and left for other processes to reuse this connection again. Else, if not defined, plugin will create a new user session every time and will destroy this after each run.	Optional
--getPerfStats	If this flag is specified this will provide performance data for the Nagios to use to plot graphs. This flag can only be used when -a option is used.	Optional

`--version`	If specified it will print the current Cisco IMC Monitoring plugin version. **NOTE**: All the other options will be ignored.	Optional
`-h/--help`	Prints the help content and the plugin input arguments supported	Optional

Table 3 : Plugin Argument Parameters

There are multiple ways in which this script can work. For example in a conventional way, user can provide a range for warning or critical values and based on the given values the plugin script can decide the service state.

```
# cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t class
-q computeRackUnitMbTempStats -a ambientTemp -w 30 -c 50
```

Else user can also provide a regular expression as OK or CRITICAL criteria.

```
# cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t class
-q storageController -a presence -r equipped
```

By default the script uses the Cisco IMC faults as the basis for returning the service state. Here user can just pass a dn or class as query and the plugin script will return CRITICAL, WARNING or OK as per the faults found on that dn or class.

```
# cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q
sys/rack-unit-1
```

So based on the query it will fetch all the related faults and if this query has a critical fault then the plugin script will return the service as CRITICAL.

In case there is no fault in the query passed then Nagios plugin script will fetch the relevant inventory information and will display the same on the Nagios web UI or CLI.

5.2 Plugin CLI Example

Below are some examples of different CLI options that can be used for fetching different type of information and status for a given query (dn or class).

CLI (**DN as input**) – This will provide status for only the given DN

cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q sys/rack-unit-1

Output

sys/rack-unit-1:OK - Serial : FCH1752V07Y,Uuid : 71BE3660-487D-4D3F-928F-11EB1DF5D800,Model : UCSC-C240-M3L,Vendor : Cisco Systems Inc,Power(W) : on,Memory(MB) : 196608,Cores : 16,CPUs : 2

CLI (**Class as input**) – This will provide the status for all the chassis in given IMC domain.

cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t class -q "EquipmentPsu"

Output

sys/rack-unit-1/psu-2:OK - Model : UCSC-PSU-650W,Power Status : on,Serial : LIT174622NW

sys/rack-unit-1/psu-1:WARNING -

WARNING - sys/rack-unit-1/psu-1-PSU1_IOUT: Power Supply 1 current is upper non critical : Reseat or replace Power Supply

==== Fault # 1 ====

Dn : sys/rack-unit-1/psu-1/fault-F0882

Descr : PSU1_IOUT: Power Supply 1 current is upper non critical : Reseat or replace Power Supply

severity : minor

Cause : power-problem

Type : server

Created : Tue May 13 03:54:21 2014

CLI (with **–a**, **-w and –c**) – Here the end user can provide a warning and a critical value for a given attribute. Based on these inputs the plugin will return the service status as per the attribute value.

./cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q sys/rack-unit-1/board/temp-stats -a AmbientTemp -w 18 -c 30

Output

WARNING - sys/rack-unit-1/board/temp-stats - AmbientTemp : 19.0

CLI (with **–a and –r**) – The end user can provide a regular expression for a given attribute and based on these inputs the plugin will return if the service status is OK or in a CRITICAL state.

./cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q sys/rack-unit-1/board/storage-SAS-SLOT-HBA/pd-1 -a online -r true

Output

CRITICAL - sys/rack-unit-1/board/storage-SAS-SLOT-HBA/pd-1 - online: false

CLI (with **-a** and **--getPerfStats**) – The end user can provide the getPerfStats flag with attribute option. When this flag is set then the CLI will return the performance data appended to the other output via a pipeline "|".

./cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q sys/rack-unit-1/board/temp-stats -a AmbientTemp --getPerfStats

Output

OK - sys/rack-unit-1/board/temp-stats - AmbientTemp : 21.0**|AmbientTemp=21.0**

CLI (with **–a, -w, -c** and **--getPerfStats**) – Here the end user can provide a warning and a critical value for a given attribute. Based on these inputs the plugin will return the service status as per the attribute value. With getPerfStats flag the attribute value and the warning and critical values are used to return the performance data.

./cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t dn -q sys/rack-unit-1/board/temp-stats -a AmbientTemp -w 25 -c 30 --getPerfStats

Output

OK-sys/rack-unit-1/board/temp-stats-AmbientTemp:21.0|**AmbientTemp=21.0;25.0;30.0;;;**

CLI (with **--inHierarchical**) – This will provide detailed view of health status for the given query

cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t class -q computeRackUnit --inHierarchical

Output

Overall Health Status - CRITICAL -

sys/rack-unit-1:CRITICAL -

CRITICAL - sys/rack-unit-1-PS_RDNDNT_MODE: Power Supply redundancy is lost : Reseat or replace Power Supply

CLI (**with –inHierarchical** and **--faultDetails**) – This will fetch all the components which are faulty (WARNING or CRITICAL) with their fault details. And will display state OK with inventory information for the parent class if no fault has been found in its hierarchy.

cisco_imc_nagios -u <username> -p <password> -H <IMC IP/FQDN> -t class -q computeRackUnit --inHierarchical --faultDetails

Output

sys/rack-unit-1:CRITICAL -

CRITICAL - sys/rack-unit-1 : PS_RDNDNT_MODE: Power Supply redundancy is lost : Reseat or replace Power Supply

==== Fault # 1 ====

Dn : sys/rack-unit-1/fault-F0743

Descr : PS_RDNDNT_MODE: Power Supply redundancy is lost : Reseat or replace Power Supply

severity : major

Cause : psu-redundancy-fail

Type : server

5.3 Default Plugin Config

```
======================================================================
# User can provide the class id and a comma separated list
# of attributes which user wants to display or check status for
# For fetching Inventory attributes from the class user needs to
# provide "Inv_" as prefix followed by the class name.

Inv_ComputeRackUnit=Serial,Uuid,Model,Vendor,OperPower:Power(W),TotalMemory:Memory(MB),
NumOfCores:Cores,NumOfCpus:CPUs
Inv_ComputeRackUnitMbTempStats = AmbientTemp:Temperature
Inv_ComputeMbPowerStats= ConsumedPower:Power(W)
Inv_ProcessorUnit= Cores, Model , Speed:CPU Speed(Mhz)
Inv_MemoryUnit=Capacity:Memory Size (MB)
Inv_MemoryArray= CurrCapacity:Total Memory (MB), Populated:Slot(s) Populated
Inv_EquipmentPsu = Model,Power:Power Status,Serial

# Define what all properties user wants from the Fault class
FaultInst=Dn,Descr,severity,Cause,Type,Created


# For fetching Statistics attributes from the class user needs to
# provide "Stats_" as prefix followed by the class name.
# User can provide the class id and list of attributes which
# user wants to fetch as part of statistics.

#Stats_<Class Name>= <Attribute_Name>;<UOM>;<warn>;<crit>;<min>;<max>
#Stats_ProcessorUnit=Speed

#User can append more "Unit Of Measurments" which they want to allow in getting preformance
statistics.
STATS_UOM_LIST =%,s,us,ms,c,B,KB,MB,TB

#User defined mapping for skipping of the faluts
#SKIP_FAULT_LIST=<Attribute Name>:<Value>,<Attribute Name>:<Value>...
SKIP_FAULT_LIST=Lc:suppressed,Type:fsm,Severity:info,Severity:condition
======================================================================
```

6. Auto Discovery Addon

6.1 Working with Auto Discovery

Currently auto discovery addon creates host and services in the Nagios system as per the details provide in the table below.

Host	Services	Service Details
IMC Host	Ping IMC	This service will check the ping to the given IMC host
	Check Rack Server	This service checks all the faults that may have occurred on the IMC host and will return Nagios state accordingly. If there are no faults on the IMC host then it will display the inventory information for the same.

Table 4 : Host and Service Mapping

The auto-discovery addon script can either be manually invoked or can be added to a cron job for periodic inventory checks. This script can take the following optional inputs from the end user.

Argument Name	Description
-f / -- csvFile	User can provide an absolute path with the IMC host csv file name. If this option is not provided then by default the path is taken as the current working directory and filename is IMCHostInfo.csv
-r /--restartService	This option provides end user with the flexibility of restarting the Nagios service after the auto discovery is finished. The default is not to restart the Nagios service. If user wants to restart the service, he can just pass this option in the CLI.
-H / --Host	IP/FQDN of an IMC domain. **Note**: If this option is given the script will skip the CSV file and only discover the Host provided by this option.
-u / --user	IMC domain login user name.
-p / --password	Specifies IMC user's password to login to the server.
-n / --port	This specifies the IMC Manager http/https port.
--NoSsl	If defined this flag will turn off the secure communication (SSL) with the IMC domain.
--proxy	User can specifies a proxy url that contains proxy connection and optionally authentication information.
-D/--disable-hostgroup-creation	This option provides user a way to disable default host groups creation via auto-discovery.
--debug	If defined it will print the xml communication between the plugin and IMC. It is also helpful for detailed debugging in case of any error that may have occurred while using this CLI. Use this for debug purpose only.
--version	If specified it will print the current Cisco IMC Auto Discovery version.' NOTE: All the other options will be ignored.
-h/--help	Prints the help content and the cli input arguments supported.

Table 5 : Auto discovery CLI options

Argument Name	Description
REMOVE_OLD_DISCOVERY	This flag in the configuration file sets the add-on script behavior to either remove old discovered device or keep old discoveries and update only the new IMC domains. If this flag is set to False, then addon will not remove the previously discovered objects. If this flag is set to True, then the old discoveries will be removed. This is the default behavior.

Table 6 : Auto discovery CFG file options

If the script is invoked without using the "-r /--restartService" cli options then, at the end of the discovery process it will prompt the user for input on restarting the Nagios service.

6.2 Add Service

As an advance feature in the auto discovery add-on user can define his own services around the rack server by editing the 'NagiosAutoDiscoveryIMC.cfg' and updating the SERVICE_LIST variable.
Here user can provide his own service name and service class or dn. Optionally, user can also provide various cli options that user want to pass to the monitoring plugin script.
This variable should be in the following format

```
<service name>:<class or dn> ,<service name>:<class or dn>:<optional cli options>
```

For example we can update the NagiosAutoDiscoveryIMC.cfg with the following custom service list like

```
SERVICE_LIST= Fault Status:ComputeRackUnit:"--inHierarchical --onlyFaults",Processors:processorUnit,Memory:memoryArray,Adaptor:adaptorUnit
```

Now when the auto discovery is executed again the following list of services will appear in the Nagios web UI

10.29.131.105	Adaptor	OK	05-14-2014 01:44:27	1d 16h 22m 18s	1/4	sys/rack-unit-1/adaptor-1:OK - Dn : sys/rack-unit-1/adaptor-1,Description : ,CimcManagementEnabled : no,Serial : FCH165076XF,Presence : equipped,Status : None,AdminState : policy,PciSlot : 1,PciAddr : 32,Vendor : Cisco Systems Inc,Model : UCSC-PCIE-CSC-02,Id : 1
	Fault Status	CRITICAL	05-14-2014 01:45:44	1d 16h 26m 1s	4/4	sys/rack-unit-1:CRITICAL - CRITICAL - sys/rack-unit-1-PS_RDNDNT_MODE: Power Supply redundancy is lost : Reseat or replace Power Supply
	Memory	OK	05-14-2014 01:45:18	1d 16h 21m 27s	1/4	sys/rack-unit-1/board/memarray-1:OK - Total Memory (MB) : 98304,Slot(s) Populated : 12
	Ping IMC	OK	05-14-2014 01:44:58	1d 16h 21m 47s	1/4	PING OK - Packet loss = 0%, RTA = 1.69 ms
	Processors	OK	05-14-2014 01:41:09	1d 16h 25m 36s	1/4	sys/rack-unit-1/board/cpu-2:OK - Cores : 6, Model : Intel(R) Xeon(R) CPU E5-2430L 0 @ 2.00GHz,CPU Speed(Mhz) : 2000 sys/rack-unit-1/board/cpu-1:OK - Cores : 6, Model : Intel(R) Xeon(R) CPU E5-2430L 0 @ 2.00GHz,CPU Speed(Mhz) : 2000

Figure 4 : Custom Service

7. Customizing Monitoring Plugin

7.1 Customize Inventory Information

As an advance feature there is a provision wherein user can select the required valid attribute(s) for a given class and provide a user friendly name to these attributes that user wants to see on the Nagios Web UI. This can be configured via the configuration file "`cisco_imc_nagios.cfg`" which is found in the same location as that of the monitoring plugin.

For fetching Inventory attributes from the class user needs to provide "Inv_" as prefix followed by the class name as the variable name and the list of attributes as its value.

So the property string should be of the following type

```
Inv_<class id >=<AttributeName>,<AttributeName>:<UserGivenName>,<AttributeName>
```

For example if the attribute name is say '`OperPower`' and user wants that to be seen as say '`Power(W)`', then user can input the following:-

```
OperPower:Power(W)
```

A complete example for a class '`ComputeRackUnit`' will look like

```
Inv_ComputeRackUnit=Serial,Uuid,Model,Vendor,OperPower:Power(W),TotalMemory:Memo
ry(MB),NumOfCores:Cores,NumOfCpus:CPUs
```

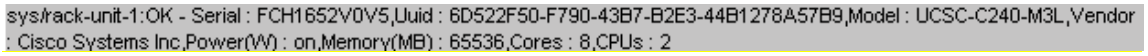

Figure 5 : Custom inventory information

7.2 Customize Statistics Information

The plugin provides user with flexibility to select the required valid attribute(s) for a given class to be used as performance data. This can be configured via the configuration file "cisco_imc_nagios.cfg" which is found in the same location as that of the monitoring plugin.

One could add entries in this configuration file for getting performance stats for specific "Class" by adding at least one of its attribute.

The basic format of the entry is as shown below:

Stats_<Class Name>= <Attribute_Name>;<UOM>;<warn>;<crit>;<min>;<max>

<Class Name>: It's the name of the class for which the stats need to be generated. The plugin will look for this entry and read the given parameters.

<Attribute Name>: This will be one of the valid attribute from the selected class. This attribute should return a numeric value as graphs are plotted against the numeric values only.
One could also give an optional name to this attribute by writing this name after ":". If this optional name is given then this will be shown as the label instead of the "attribute name". Below is an example for it.

Stats_MemoryArray=CurrCapacity:"Current Capacity(MBs)"
<class name> = <attribute name> : <attribute optional name>

<UOM>: Unit of measurement. It's the unit associated to the value of the attribute. This field is OPTIONAL and can be left blank. It can have one of the following values.

a. no unit specified - assume a number (int or float) of things (eg, users, processes, load averages)
b. s - seconds (also us, ms)
c. % - percentage
d. B - bytes (also KB, MB, TB)
e. c - a continuous counter (such as bytes transmitted on an interface)

Note:

Allowed Unit of measurement is controlled by "STATS_UOM_LIST" parameter present in configuration file. User can update the list according to the use.

#User can append more "Unit Of Measurements" which they want to allow in getting performance statistics.
STATS_UOM_LIST =%,s,us,ms,c,B,KB,MB,TB

<Warn>,<Crit>,<Min>,<Max> : These are OPTIONAL parameters. These parameters can either be a numeric value or a parameter of the class returning some numeric number. All these parameters should be in same "Unit of Measurement" as that of label. If any of these value is not present or user does not want to set any parameter for them then he can leave these fields blank.

Warn – It sets the warning threshold in graphing the stats for that attribute.
Crit – it sets the Critical level threshold in graphing the stats for that attribute.
Min – This field sets the minimum possible value for the selected attribute.
Max – This field sets the maximum possible value for the selected attribute.

Below are few possible ways to write Stats Class definition in the configuration file.

Only attribute defined all optional parameters skipped.
Stats_ProcessorUnit=Speed

#Optional name for attribute has been mentioned.
Stats_ProcessorUnit=Speed:CPU_Speed(Mhz)

#Few optional parameter skipped
Stats_ComputeRackUnitMbTempStats= AmbientTemp;;;;18;34

#All optional parameters provided.
*Stats*_ComputeRackUnitMbTempStats=AmbientTemp:Temp;c;25;30;18;34

Now when Nagios service is called and uses one of these Stats "class", then with the normal inventory related data, the plugin will also return the listed attribute as performance stats.

Below is the CLI output of a service call:
./cisco_imc_nagios - u <username> -p <password> -H <IMC IP/FQDN> -t class -q ComputeRackUnitMbTempStats

sys/rack-unit-1/board/temp-stats:OK – Temp : 22.0**|Temp=22.0c;25;30;18;34;**
Here the attribute "Temp" after '|' is the performance stat for this service. When such a service is run in Nagios, then this performance stat is stored in historical information database which then a third party graphing tool uses to populate graphs.

On Nagios GUI “Performance Data” field in the service gets populated when this service is run.

Service State Information

Current Status:	OK (for 0d 0h 4m 54s)
Status Information:	sys/rack-unit-1/board/temp-stats:OK - Temperature : 21.0
Performance Data:	AmbientTemp=21.0c;25;30;18;34;
Current Attempt:	1/4 (HARD state)
Last Check Time:	2015-10-07 16:34:58

Figure 6 : Performance Data

Note:

- The plugin follows Nagios generic guidelines for generating performance data.
- User can install any third party graphing plugin from Nagios Communities to populate graphs by using performance data.

Figure 7 : Graph plotted using performance data

7.3 Customize Fault Information

User can also control the attributes that are seen in the fault details by editing `'cisco_imc_nagios.cfg'` and updating the `‘FaultInst’` variable with the required attribute names.

So for example user can have the following list of attributes:-
`FaultInst=Dn,Descr,severity,Cause,Type,Created`

```
==== Fault # 1 ====
Dn : sys/rack-unit-1/board/storage-SAS-SLOT-MEZZ/fault-F1003
Descr : Storage controller SLOT-MEZZ patrol read failed: Patrol Read can't be started.
severity : warning
Cause : equipment-inoperable
Type : server
Created : Mon Apr 21 22:34:32 2014
```

Figure 8 : Custom fault details

7.4 Skipping Faults

With this feature user can define the faults user wants to skip from the Nagios monitoring service.

For example, if user wants to skip faults with severity as ‘info’ then this can be defined as the value for the configuration variable SKIP_FAULT_LIST in the monitoring plugin configuration file 'cisco_imc_nagios.cfg'.

The format of value for this variable is of type

```
<fault attribute>:<value>,<fault attribute>:<value>.
```

Example:

```
SKIP_FAULT_LIST=Lc:suppressed,Type:fsm,Severity:info
```

This will skip faults which are either marked suppressed or whose severity is info.

Note:

Skipping fault based on “Description” field is not advisable as it might contain some special characters which might not let the fault to be skipped when a comparison is done.

8. Uninstalling the Solution

Uninstallation process requires Nagios services to be stopped first. Uninstaller will ask for confirmation before trying to stop Nagios service.

To uninstall the Cisco IMC Nagios integration, follow the step as mentioned below

Run the `installer.py` with '`--uninstall`' option

```
# ./installer.py --uninstall
```

9. Known Caveats

In case user password contains any special character then it has to be provided in double quotes.

Example

192.168.1.16,admin,"pass,word"
192.168.1.16,admin,"My_password"