MONITORING EMC GREENPLUM DCA WITH NAGIOS

EMC Greenplum Data Computing Appliance, EMC DCA Nagios Plug-In, Nagios Core Infrastructure Monitoring

- Monitor DCA hardware components
- Monitor DCA database and Hadoop services
- View full DCA health in Nagios display

Abstract

This white paper describes a software plug-in that can be used with the Nagios Core infrastructure monitoring framework to display the state of a wide range of components in the EMC® Greenplum™ Data Computing Appliance (DCA).

May 2012
# Table of contents

**Executive summary** ................................................................. 5  
  Business case ............................................................................. 5  
  Solution overview ...................................................................... 5  
  Key results ................................................................................. 5  

**Introduction** ........................................................................ 6  
  Purpose ....................................................................................... 6  
  Scope ......................................................................................... 6  
  Audience .................................................................................... 6  
  Terminology ............................................................................... 6  

**Technology overview** ............................................................ 8  
  Introduction ............................................................................... 8  
  EMC Greenplum Data Computing Appliance ............................. 8  
  Nagios ....................................................................................... 9  
    Nagios Core ........................................................................... 9  
    EMC DCA Nagios plug-in ...................................................... 10  

**Greenplum DCA SNMP MIB** .................................................. 11  
  Introduction ............................................................................. 11  
    MIB browser .......................................................................... 15  

**Nagios Core** ......................................................................... 17  
  Introduction ............................................................................. 17  
  Service checks and commands ............................................. 19  

**DCA Nagios plug-in** ............................................................... 20  
  Introduction ............................................................................. 20  
  Open source software ........................................................... 20  
  Accessing the plug-in on Nagios .......................................... 20  
  Service checks ......................................................................... 20  
  DCA Nagios plug-in command line ...................................... 24  
    Example 1 - Running the Greenplum Database version on the DCA cluster ...................... 25  
    Example 2 - Checking the first Segment Server's fan status ........................................... 25  
    Example 3 - Checking the RAM in the Master .................................................................. 26  
  Plug-in command objects ...................................................... 26  
  Deploying the DCA Nagios plug-in ........................................ 29  

**Conclusion** .......................................................................... 31  
  Summary .................................................................................... 31
References ....................................................................................................................................... 32

White papers .................................................................................................................................. 32
Other documentation....................................................................................................................... 32
Executive summary

Business case
The EMC® Greenplum™ Data Computing Appliance (DCA) is a combination of servers, switches, and software that together form an extremely powerful analytics platform.

As businesses use this platform for increasingly critical business intelligence operations, it is important to be able to monitor the state of the appliance to ensure it is operating at peak efficiency. To enable you to integrate the EMC Greenplum DCA into standard data center monitoring frameworks, the DCA supports the Simple Network Management Protocol (SNMP), which can be used to determine the state of various components in the DCA.

Solution overview
Nagios is open-source network management software (NMS) that is used to monitor the state of data center equipment and to display that information in a user interface. Nagios can also generate alerts if monitored values exceed specified thresholds.

Nagios provides an extensible architecture, based on plug-ins, that makes it easy to add monitoring capabilities to different types of data center equipment.

EMC has developed a Nagios plug-in for the Greenplum DCA that allows you to build a comprehensive monitoring and alerting application.

Note
EMC also provides Greenplum Command Center, which is a web-based application that provides advanced monitoring and management capabilities for the EMC Greenplum Database™ and Hadoop modules and associated hardware.

Key results
The DCA Nagios plug-in enables you to monitor the Greenplum DCA and detect problems before and as they occur. This enables you to avoid or reduce system downtime and simplifies your monitoring operations.
Introduction

Purpose
This white paper provides an overview of the DCA SNMP management information base (MIB), describes the operation of the DCA Nagios plug-in, and shows examples of how you can configure Nagios to use the plug-in and monitor the state of one or more DCAs.

Scope
The scope of this white paper is to:

- Provide a brief overview of the Greenplum DCA MIB
- Describe the Nagios architecture
- Describe the DCA Nagios plug-in
- Provide examples of how the Nagios plug-in can be used to cause Nagios to monitor various aspects of a DCA

Audience
This white paper is intended for a technical audience, including:

- Technical consultants who may want information about monitoring a Greenplum DCA with SNMP
- Customers who are using Nagios as a data center monitoring tool and want to integrate the Greenplum DCA into their environment
- EMC consultants who want to understand the capabilities of the DCA Nagios plug-in

This white paper assumes you have a basic understanding of Nagios running on a Linux system.

Terminology
This paper includes the following terminology.

Table 1. Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNMP agent</td>
<td>Software that runs on a device that responds to SNMP queries from network management software. Agents may also accept commands to control the device. Agents may also asynchronously generate messages or traps without being specifically queried.</td>
</tr>
<tr>
<td>SNMP network management software (NMS)</td>
<td>A software package that you can configure to query SNMP agents on various devices to determine their status. An NMS package will also usually accept SNMP traps sent from SNMP agents without being queried. NMS packages have a provision for generating alerts (email or text message, for example) if the state of monitored equipment meets specified criteria (temperature or error count, for example).</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Management information base (MIB)</td>
<td>The MIB defines and describes the kinds of messages that an SNMP agent can generate and how the NMS can query the agent.</td>
</tr>
<tr>
<td>Object identifier (OID)</td>
<td>An OID is the definition of an individual device or component in the MIB. An example of an OID might be “Power Supply State”. The OID describes how to query the component and what kind of response is returned. There are typically tens or hundreds (if not thousands) of OIDs defined in a MIB in a hierarchical structure.</td>
</tr>
<tr>
<td>SNMP trap</td>
<td>A trap is a message that an SNMP agent sends when an event happens that the agent is monitoring. An SNMP agent responds to queries from the NMS but may also be able to send data to the NMS without being queried. This unsolicited message is called a trap.</td>
</tr>
</tbody>
</table>
Technology overview

Introduction

This chapter provides an overview of the key components of the solution, which include:

- EMC Greenplum Data Computing Appliance
- Nagios data center monitoring software
- Nagios plug-in for EMC Greenplum DCA

EMC Greenplum Data Computing Appliance

EMC Greenplum Data Computing Appliance (DCA) is a purpose-built appliance that delivers a fast-loading, highly scalable, and parallel-computing platform for next-generation data warehousing and analytics. The appliance architecturally integrates database, computing, storage, and network resources into an enterprise-class, easy-to-implement system. The Greenplum DCA offers the power of a massively parallel processing (MPP) architecture, delivers the fastest data loading capacity in the industry, and provides the best price-to-performance ratio without the complexity and constraints of proprietary hardware.

Figure 1 shows the modular approach of the Greenplum DCA; "GPDB" refers to Greenplum Database, "HD" identifies a Hadoop module, and "DIA" identifies a data integration accelerator.

Figure 1. EMC Greenplum DCA modules

For more information about the Greenplum DCA, see the white paper EMC Greenplum Data Computing Appliance: Architecture, Performance, and Functions.
Nagios

Nagios is an open-source IT infrastructure-monitoring application that helps organizations identify and resolve IT infrastructure problems before they affect critical business processes. Nagios can monitor data center equipment and processes to ensure that they are functioning properly. Nagios can alert you of problems or potential problems, which enables you to begin remediation processes before an outage affects your users.

Nagios is built on a plug-in architecture that enables it to be extended so you can monitor any type of equipment in the data center. Plug-ins that understand how to communicate with a particular device can be added to a Nagios configuration, and information from that device is displayed on the Nagios interface. Figure 2 provides an overview of the Nagios architecture.

Figure 2. Nagios architecture

Nagios Core

Nagios includes several product tiers:

- Nagios XI
- Nagios Fusion
- Nagios Core

This white paper describes only the Nagios Core product.
Nagios includes a core set of plug-ins that can monitor many types of common data center equipment and processes. To monitor specific types of equipment, such as the Greenplum DCA, custom plug-ins can be written.

EMC has written a Perl program, the DCA Nagios plug-in, that enables Nagios to monitor the status of the Greenplum DCA. The DCA Nagios plug-in uses SNMP to communicate with the Greenplum DCA Master node and retrieve status information about various components of the Greenplum DCA.

The DCA Nagios plug-in depends on the enhanced Greenplum DCA SNMP MIB that was introduced in Greenplum DCA software 1.2.0.0 and Greenplum Database 4.2.1.
Greenplum DCA SNMP MIB

Introduction

This chapter provides a brief overview of the Greenplum DCA SNMP MIB and describes what kinds of information you can obtain from the Greenplum DCA by using the MIB.

The enhanced SNMP MIB and agent scheme that was introduced with Greenplum DCA software 1.2.0.0 allows the entire Greenplum DCA cluster to be monitored by sending an SNMP query to the Master node. In this scheme, each component of the Greenplum DCA – Master Servers, Segment Servers, Hadoop nodes, DIA nodes, and switches – communicates internally and feeds SNMP status information back to a master SNMP agent (running on the Master node). An NMS (such as Nagios) can then send SNMP queries to the Greenplum DCA Master to obtain the status of segments, switches, and so on. The NMS does not contact individual components of the Greenplum DCA directly but instead sends SNMP queries to the Greenplum DCA Master to ask for the status of a particular component.

Note The Greenplum DCA SNMP agent also has the ability to send SNMP traps to the NMS. Trap handling is not available in the current DCA Nagios plug-in.

In SNMP, the definition of what information you can obtain through SNMP queries is described in the SNMP MIB. The Greenplum DCA MIB defines which components you can query and what kinds of information you can obtain from the Greenplum DCA. The MIB is built as a nested structure of object identifiers (OIDs); each OID describes the component that is being referenced and how it relays the information.

Figure 3 shows a high-level view of the Greenplum DCA MIB, as shown in a MIB browser. A MIB browser helps you to understand the structure and components that you can query.

```
[ gpDCAPublic ]
  [ gpDCAData ]
    [ gpDCAv1Hardware ]
    [ gpDCAv2Hardware ]
    [ gpDCAServices ]
    [ gpDCAIsoVersion ]
    [ gpDCAHadoopVersion ]
```

Figure 3. DCA MIB

Note The gpDCAv2Hardware OID is a placeholder to accommodate potential future hardware versions of the Greenplum DCA. This OID currently does not return any information.

The MIB is separated into hardware and services trees and also includes version information. The gpDCAIsoVersion is an example of an OID that you can query through SNMP. Figure 4 shows the definition of this OID.
The definition shows that the OID returns a string that indicates the version of software on the Greenplum DCA cluster.

Figure 5 shows the gpDCAv1Hardware tree expanded so you can view the sub-trees for Master nodes, Segment nodes, admin switches, interconnect switches, and so on.

You can expand each of the hardware OIDs further, as shown in Figure 6.
Figure 6. Expanded hardware tree

The expanded gpMasterNodes tree shows that a table of masterNodeEntry OIDs is available. A table in a MIB indicates that multiple instances of the structure may be present. In this case, the gpMasterNodeTable contains instances for both the Master and Standby Master nodes.

The gpMasterHostNumber OID and the gpMasterHostName OID indicate which Master (1 or 2) is responding and what its name is (mdw or smdw). As shown in Figure 6, the column of OIDs is like a row in the Master Node table; there is one row for the GP Master node and one row for the GP Standby Master node.
Some columns have nested structures. For example, as shown in Figure 7, if you expand the `gpMasterVirtualDisk OID`, there are three virtual disks; each has a set of OIDs with information about the disk.

![Sample of nested structure](image)

**Figure 7. Sample of nested structure**

This type of structure exists throughout the Greenplum DCA MIB, so when you query the Greenplum DCA for a certain piece of information, the process consists of drilling through the MIB structure to find the OID you want and building an SNMP query that asks for that particular OID.

As shown in Figure 8, each OID is actually a dotted decimal sequence that uniquely identifies the OID.
Monitoring EMC Greenplum DCA with Nagios

EMC Greenplum Data Computing Appliance, EMC DCA Nagios Plug-In, Nagios Core Infrastructure Monitoring

15

Monitoring EMC Greenplum DCA with Nagios

EMC Greenplum Data Computing Appliance, EMC DCA Nagios Plug-In, Nagios Core Infrastructure Monitoring

This particular OID returns an integer, with values from 1 to 5, that indicates the status of the disk. The OID value indicates the OID hierarchical structure and is the value used in the SNMP query that is sent to the Greenplum DCA to obtain the corresponding status value.

**Note**

OIDs that are contained within MIB tables (as is the case here) require that you append an extra index value to the OID value shown in order to index to the correct row in the MIB table.

For example, to obtain the value for gpMasterVirtualDiskStatus3, you would nominally send the OID value listed (the one ending in .1.1.13.3.1). However, since this OID is in the gpMasterNodeTable, you must indicate the table row you want to access (in effect, whether you want information from the Master or the Standby Master) and, therefore, you must add a digit to the end of the OID you send. Thus, to access the status on the Master node, you would send an OID ending in .1.1.13.3.1.1.

A full explanation of the Greenplum DCA MIB is beyond the scope of this white paper, but it is important that you have an understanding of the structure and how to use SNMP to access individual OIDs.

**MIB browser**

One of the best ways that you can explore a MIB and experiment with SNMP queries is to use a MIB browser. The MIB browser from iReasoning not only makes it easy to navigate the structure of the MIB but also generates the SNMP queries and returns the values you request.

You should install the MIB browser on a client machine that has a connection to the DCA Master, and load the Greenplum DCA MIB definition file into the browser. You can access the Greenplum DCA MIB file on the Greenplum DCA in /usr/share/snmp/mibs in the file GP-DCA-DATA-MIB.txt.
After you set the IP address of the DCA Master in the browser, the browser displays values or trees of values of the OIDs you specify, as shown in Figure 9.
Nagios Core

Introduction

Nagios Core is a popular software package for monitoring data center equipment and is available for download from the Nagios web site.

A full explanation of Nagios is beyond the scope of this white paper, but a good reference is the book *Nagios, System and Network Monitoring* by Wolfgang Barth. The Nagios web site also has documentation, installation guides, and forums that are very helpful in getting started.

Nagios Core consists of monitoring and alerting functions, as well as a web application that takes care of displaying the data center information that is being monitored in a web browser. Once Nagios Core is installed, you open a web browser and point it to the Nagios server. You are prompted to log in with the username and password that were configured during installation. The Nagios Core home page is displayed, as shown in Figure 10.

![Figure 10. Nagios Core home page](image)

On the home page, the links in the left pane take you to the monitoring windows.

Nagios Core monitors hosts and services. Host monitoring – called a host check – determines whether a host is “up” or not. A typical host check involves sending a “ping” command to the host to see if it returns an answer.
Service monitoring – called service checks – determines whether certain services, such as httpd, smtp, print queues, and so on, are running on a host. Service checks are the heart of the Nagios Core monitoring scheme and are implemented in the Greenplum DCA Nagios plug-in.

Nagios Core itself is not capable of checking to see if a particular service is running; instead, it provides a framework for you to install a Nagios plug-in that does know how to monitor that service. After you install the plug-in, you can configure Nagios Core to call that plug-in on a periodic basis to check the service. The plug-in returns data in a particular format, and Nagios Core displays that information in the monitoring dashboard.

Nagios Core includes a set of standard plug-ins that can check a variety of popular services that you may want to monitor. Figure 11 shows a set of services being monitored on the Nagios server.

The entries in the Service column correspond to one or more plug-ins that are executed by Nagios Core to determine the state or value of a service. The plug-in returns a status code that Nagios interprets as either “Okay”, “Warning”, “Error”, or “Unknown”. The plug-in can also return descriptive text that is displayed in the Status Information column.

You can also pass parameters to the plug-in that control how the plug-in behaves when it runs. Typical parameters are values that specify thresholds for warning and error conditions.

For example, parameters that indicate that free disk space below 1 GB is a “Warning” condition and free disk space below 50 MB is a “Critical” condition might be passed to a plug-in that checks available disk space. When the plug-in checks the available disk space, it uses these threshold values to determine whether to return an “Okay”, “Warning”, or “Error” status code based on the free disk space it sees. Using command line parameters, a single plug-in can perform many different operations.

The DCA Nagios plug-in accepts different parameters that let it check all the different OIDs in the Greenplum DCA MIB.
A Nagios service check is an object you define in Nagios Core to call a plug-in. For example, you may want to define a service check that looks at the battery state of the Master. In a Nagios configuration file, you would define this service check by giving it a service description and telling it which plug-in to call.

This Nagios service check definition example checks the battery status:

```plaintext
define service{
    use   dca-master-service
    service_description  Master Battery
    check_command check_dca_nodes_snmp!Battery!Master!1
}
```

The `use` directive specifies a service template, defined elsewhere, that specifies generic service check properties, such as polling frequency and alerting options. The `service_description` value provides the name of the service check – Master Battery – and the `check_command` indicates the command object to call and the parameters (separated with exclamation marks) to pass to it.

The command object is a reference to an object defined in Nagios that defines how to call a plug-in. This example defines a command object named `check_dca_nodes_snmp`, which is referenced in the Master Battery service check. This calls the DCA Nagios plug-in:

```plaintext
define command{
    command_name check_dca_nodes_snmp
    command_line $USER1$/check_snmp_dca.pl -H $HOSTADDRESS$ -o '
    $ARG1$' -n $ARG2$ -i $ARG3$ -w $ARG4$ -c $ARG5$
}
```

The `command_line` directive defines the command line that Nagios uses to execute the plug-in. Nagios command objects make extensive use of macros that take on values dynamically based on how the command object is referenced in a service check definition.

The `$USER1$` and `$HOSTADDRESS$` macros specify the location of the plug-in directory and the name of the host that is being referenced in the service check. The `$ARGx$` macro corresponds to the parameters between exclamation marks in the service check definition that calls this command object.

In reviewing the Master Battery service check definition, you can see that three parameters were passed to the plug-in – Battery, Master, and 1. These correspond to the `$ARG1$`, `$ARG2$`, and `$ARG3$` macros that are assigned to the `-o`, `-n`, and `-i` command line arguments in the Perl script.

**Note** The `-w` and `-c` options are not given values in this case. These are optional parameters for the plug-in and are not needed in some service checks.
DCA Nagios plug-in

Introduction
A Nagios plug-in is a stand-alone program. You can execute it from the command line completely outside of the Nagios framework and therefore can develop it independently from the Nagios system. As long as the program exits with a return code that Nagios expects (a value from 0 to 4), and optionally prints additional information to Standard Out, it can function as a Nagios plug-in.

The DCA Nagios plug-in is a program written in Perl that understands how to query the Greenplum DCA through SNMP and process the responses. The plug-in accepts command line parameters that tell it which OID (or OIDs) to query, optional threshold values that indicate when to return warning or error states, and other values that control its operation. The plug-in returns a return code that Nagios uses to indicate Okay, Warning, Error, or Unknown and also prints additional information to Standard Out. Nagios displays this information in the informational area of the Nagios GUI (web page).

Open source software
The DCA Nagios plug-in is one way to enable Nagios to display and monitor the Greenplum DCA. Users are encouraged to extend and improve the plug-in and fully test its operation before deploying in a production environment.

Note: EMC does not offer support for this plug-in. It is provided “as is” to the open source community as a reference design.

Accessing the plug-in on Nagios
The DCA Nagios plug-in is available on the Nagios Exchange web site. Simply search for “Greenplum” to find the DCA Nagios plug-in and any associated plug-ins.

Service checks
Conceivably, you could write a separate plug-in for each component in the Greenplum DCA that you want to check. Given the hundreds of OIDs in the MIB however, that would be impractical. Therefore, the DCA Nagios plug-in accepts parameters that indicate which OID to check and generates the appropriate SNMP query.

It would also be impractical to define service checks for every possible OID in the Greenplum DCA MIB. Not only would there be potentially thousands of service check definitions in the Nagios configuration files, but the Nagios GUI would be filled with entries from every individual component mapped to the MIB.

When you look at the DCA MIB, you can see that many of the OIDs are related. Figure 12 shows the OIDs related to the Master node power supply.
You could define a service check for each OID across the two power supplies in the Master, but, since there are power supplies in each of the two Master nodes in the Greenplum DCA, as well as in every Segment/Hadoop node, this would result in a huge number of service check entries in the Nagios GUI.

To minimize the number of service checks to define and display, but still cover all of the OIDs in the MIB, the DCA Nagios plug-in checks higher-level Greenplum DCA objects. In the case of power supplies, a service check that checks the power supply object is defined. When the DCA Nagios plug-in is called and told to check the Power Supply object on the Standby Master, the plug-in knows to generate SNMP queries for all four OIDs for each power supply (eight OIDs in total) on the Standby Master and to consolidate the results into a composite status of both power supplies for display in the Nagios GUI. Figure 13 shows a section of the Nagios display that gives the overall state of the power supplies on the Master.

![Figure 12. Master node power supply OIDs](image)

You can click the Master Power Supplies link to get a more detailed status display that shows the responses from all of the OIDs that were queried, which comprise the Power Supply object as shown in Figure 14.

![Figure 13. Master power supplies](image)
### Service State Information

<table>
<thead>
<tr>
<th>Current Status:</th>
<th><a href="#">OK</a> (for 0d 3h 12m 6s)</th>
</tr>
</thead>
</table>
| Status Information: | OK - Power Supply  
State(1): ok  
Amps(1): 0.40  
Volts(1): 214.00  
State(2): ok  
Amps(2): 0.40  
Volts(2): 214.00 |

<table>
<thead>
<tr>
<th>Performance Data:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Attempt:</td>
</tr>
<tr>
<td>Last Check Time:</td>
</tr>
<tr>
<td>Check Type:</td>
</tr>
<tr>
<td>Check Latency / Duration:</td>
</tr>
<tr>
<td>Next Scheduled Check:</td>
</tr>
<tr>
<td>Last State Change:</td>
</tr>
<tr>
<td>Last Notification:</td>
</tr>
<tr>
<td>Is This Service Flapping?</td>
</tr>
<tr>
<td>In Scheduled Downtime?</td>
</tr>
<tr>
<td>Last Update:</td>
</tr>
</tbody>
</table>

**Figure 14. Detailed status display**

You can see a more complex example in the interface OIDs for the Interconnect switch. As shown in Figure 15, there are many ports on a switch and multiple OIDs for each port.

**Figure 15. Multiple OIDs**

An abstract interconnect interface service check is defined that, when sent to the plug-in to check, results in the plug-in querying all of the interface OIDs and consolidating the results into one composite status. If any of the individual interface OIDs indicate a warning or error, the composite status is a warning or error and the description text gives the details about which interface (or interfaces) has the actual issue.
Figure 16 shows assorted service check display states.

Clicking the service check name link provides you with more details about the service check error, as shown in Figure 17.

![Service State Information](image)

Figure 16. Service check display states

Finally, if you analyze the entire Greenplum DCA MIB and think about the types of information provided, you find that there are several main OID categories and ways in which you may want to query them. For example:

- **Threshold checks**—Some OIDs return measurement values that you can use to compare against a threshold to determine if that component is operating normally, in a warning state, or in an error state. Examples might include power supply watts, free disk space, free memory, and so on.

- **Status checks**—Some OIDs return a status value – Normal, Warning, and Error – and a text description of that state.

- **Value checks**—Some OIDs return a text value. An example is the OID that returns the version number of the Greenplum DCA software.
Note You may want to treat this as a threshold check and display a warning or error if the software version is below a certain value. The DCA Nagios plug-in can be enhanced to handle this type of check.

- Composite checks—Some OIDs are part of a tree of related components that make sense to group together and show as an aggregated status. These OIDs typically return a status value as well as a component description and a status description.

Nagios plug-ins must accept a certain number of standard command line options and then are free to accept additional options that are specific to their particular functions. Table 2 lists the standard Nagios plug-in command line options.

Table 2. DCA Nagios plug-in standard command line options

<table>
<thead>
<tr>
<th>Command line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-H &lt;hostname&gt;</td>
<td>Indicates the name or IP address of the host to query for the status. For the DCA Nagios plug-in, this corresponds to the Greenplum DCA Master.</td>
</tr>
<tr>
<td>-w &lt;warning threshold(s)&gt;</td>
<td>Indicates the value (or values) below which a warning status should be returned. You can specify multiple threshold values, separated by commas, which would be applied to multiple OIDs involved in the check.</td>
</tr>
<tr>
<td>-c &lt;critical threshold(s)&gt;</td>
<td>Indicates the value (or values) below which an error status should be returned. You can specify multiple critical values, separated by commas, which would be applied to multiple OIDs involved in the check.</td>
</tr>
<tr>
<td>--help</td>
<td>Causes the plug-in to display syntax information that describes the available command line options and what they mean. This takes the place of “man” pages for the plug-in to reduce the number of files that need to be installed with the plug-in.</td>
</tr>
</tbody>
</table>

In addition to the standard Nagios plug-in command line options, the DCA Nagios plug-in defines additional command line options, as shown in Table 3.

Table 3. DCA Nagios plug-in command line options

<table>
<thead>
<tr>
<th>Command line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n &lt;node type&gt;</td>
<td>Indicates the type of node that you want to query. Possible values are Master, Segment, Admin, Int, ETL, HDMaster, HDWorker, AggAdmin, and AggInt. Each type indicates a particular tree in the Greenplum DCA MIB.</td>
</tr>
<tr>
<td>Command line</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-i &lt;node index&gt;</td>
<td>Indicates the index of the node table that you want to query. Each node type has one or more entries in a MIB table that correspond to an instance of that node in the Greenplum DCA cluster. For example, the Master node table has two entries, one for the Master node and one for the Standby node. Similarly, the Segment node table has four or more entries that correspond to the number of Segment Servers installed in the Greenplum DCA. The node index number indicates which of these nodes you want to query.</td>
</tr>
<tr>
<td>-o &lt;object identifier&gt;</td>
<td>Indicates the abstract Greenplum DCA object you want to query. A Greenplum DCA object corresponds to either a specific Greenplum DCA OID or a collection of OIDs in the Greenplum DCA MIB. You define these OIDs in the DCA Nagios plug-in and allow the Nagios Administrator to ask for status on logical components – for example, the Interconnect switch – instead of having to specify each OID individually (Interconnect switch port 1, 2, 3, and so on).</td>
</tr>
</tbody>
</table>

Given these command line options, the following examples show different ways to call the DCA Nagios plug-in to obtain information about the Greenplum DCA.

**Example 1 - Running the Greenplum Database version on the DCA cluster**
The first example simply gets the Greenplum Database version that is running on the Greenplum DCA cluster. The -H parameter is the DNS name of the Greenplum Master. This corresponds to the external IP address of the Greenplum DCA Master. The -o parameter specifies the abstract MIB object to query. The return code from the program is not shown, but the output gives a text description of the results of the query.

```
check_snmp_dca.pl -H mydca -o "DB Version"
OK - DB Version: DB Version:4.2.1.0 build 1
```

**Example 2 - Checking the first Segment Server's fan status**
This example checks the state of the fans on the first Segment Server.

```
check_snmp_dca.pl -H mydca -o "Cooling Device" -n Segment -i 1
OK - Cooling Device
Desc(1):ok: System Board FAN MOD 1A RPM
Desc(2):ok: System Board FAN MOD 1B RPM
Desc(3):ok: System Board FAN MOD 2A RPM
Desc(4):ok: System Board FAN MOD 3A RPM
Desc(5):ok: System Board FAN MOD 3B RPM
Desc(6):ok: System Board FAN MOD 4A RPM
Desc(7):ok: System Board FAN MOD 4B RPM
Desc(8):ok: System Board FAN MOD 5B RPM
Desc(9):ok: System Board FAN MOD 5A RPM
Desc(10):ok: System Board FAN MOD 2B RPM
```
Example 3 - Checking the RAM in the Master

This example is more complicated. It checks the total and available RAM in the Master. The MIB has OIDs for both total RAM and available RAM on a node. To reduce the number of checks you have to define, you define a RAM object that covers both OIDs. You pass this object name, along with the node type and instance parameter, to the plug-in.

The total and available RAM OIDs do not return a status; they simply return the amount of RAM in bytes. To indicate warning or error conditions, you also pass warning and critical thresholds to the Nagios plug-in. You pass two values for each OID (separated by commas) to provide values for both total RAM and available RAM.

```
check_snmp_dca.pl -H 10.110.80.30 -o "RAM" -n Master -i 1 -w 49319684,37460868 -c 48000000,25000000
OK - RAM: Total:49319684 Available:47653836
```

In this example, you can see that below 49,319,684 total RAM and 37,460,868 available RAM should be a warning condition and below 48,000,000 total RAM and 25,000,000 available RAM should be a critical or error condition. The Nagios plug-in reports that the current values are above those thresholds and therefore things are okay.

If you adjust the warning levels upward, you can force a warning response. In the example below, we adjusted the warning level to just above the total RAM, so the Nagios plug-in warns that the OIDs are below that level.

```
check_snmp_dca.pl -H 10.110.80.30 -o "RAM" -n Master -i 1 -w 49419684,37460868 -c 48000000,25000000
WARNING - RAM: Total:49319684 < 49419684 Available:47653184
```

Table 4 lists the command objects that can be passed to the DCA Nagios plug-in.

<table>
<thead>
<tr>
<th>Object name</th>
<th>Node type</th>
<th>Index range</th>
<th>Warn/critical options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accesses read/write</td>
<td>Master</td>
<td>1,2</td>
<td>Warn above, critical above</td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1,{number of segments}</td>
<td></td>
</tr>
<tr>
<td>Admin switch interface</td>
<td>Admin, AggAdmin</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Admin switch status</td>
<td>Admin, AggAdmin</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Array disk</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1,{number of segments}</td>
<td></td>
</tr>
<tr>
<td>Battery</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td>Object name</td>
<td>Node type</td>
<td>Index range</td>
<td>Warn/critical options</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------------</td>
<td>-------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Bytes read/write</td>
<td>Master</td>
<td>1,2</td>
<td>Warn above, critical above</td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Cache device</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Change tracking status</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Connection status</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Controller</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Controller battery</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Cooling device</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Core file status</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>CPU percent</td>
<td>Master</td>
<td>1,2</td>
<td>Warn above, critical above</td>
</tr>
<tr>
<td></td>
<td>Segment, ETL, HDMaster, HDWorker</td>
<td>1, number of segments</td>
<td></td>
</tr>
<tr>
<td>Crash file status</td>
<td>Master</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td>Object name</td>
<td>Node type</td>
<td>Index range</td>
<td>Warn/critical options</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>----------------------------</td>
<td>------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Database version</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Disk used</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Disk used percentage</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Interconnect interface</td>
<td>Int, AggInt</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Int, AggInterconnect sensor</td>
<td>Int, AggInt</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Int, AggInterconnect switch status</td>
<td>Int, AggInt</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Master mirror sync status</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Memory device</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Network device</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>OS memory</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Overall status</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
<tr>
<td>Power supply</td>
<td>Master</td>
<td>1,2</td>
<td>-</td>
</tr>
</tbody>
</table>
Monitoring EMC Greenplum DCA with Nagios

EMC Greenplum Data Computing Appliance, EMC DCA Nagios Plug-In, Nagios Core Infrastructure Monitoring

## Deploying the DCA Nagios plug-in

The DCA Nagios plug-in package consists of the plug-in itself and a set of sample Nagios configuration files that you can use to get started monitoring a Greenplum DCA.

Place the Perl script `check_dca_snmp.pl` in the `/usr/local/nagios/libexec` directory or wherever Nagios plug-ins are located in your Nagios installation. The plug-in requires
that the Perl Net::SNMP module is installed on the Nagios server. For instructions on how to install Net::SNMP on your Nagios system, see the Perl repository at CPAN.

The configuration files contain service check definitions, command definitions, and corresponding Nagios objects that show how you can call the DCA Nagios plug-in, as well as how you can structure the service checks to display in the Nagios GUI.

Since the service check definitions can get lengthy, the configuration files are separated into functional areas.

For example, there is a master.cfg file, a switches.cfg file, and a segmentxx.template file. The segmentxx.template file is a template for creating a number of segmentxx.cfg files that correspond to each segment in your system (segment01.cfg, segment02.cfg, and so on).

If you have Hadoop and DIA nodes or just DIA nodes in your Greenplum DCA system, you need to create configuration files for these nodes as well. These configuration files would be similar to the configuration files for the Segments. Place these files in a subdirectory under the /usr/local/nagios/etc directory (or equivalent). In the nagios.cfg file in that directory, you can specify the directory that holds the configuration files.

For example, you can create the following definition, which causes Nagios to look in that directory for the configuration files:

cfg_dir=/usr/local/nagios/etc/dca
Conclusion

Summary

EMC Greenplum DCA is an analytics platform that provides mission-critical information to today’s businesses. The Greenplum DCA SNMP MIB provides a wealth of information about the Greenplum DCA to ensure that it is operating at peak efficiency.

Nagios is a popular system monitoring framework that is extensible to enable nearly any kind of data center equipment to be monitored. The DCA Nagios plug-in gives you the ability to monitor the Greenplum DCA, display status information in the Nagios GUI, and generate alerts if problems are detected.
References

White papers
For additional information, see this white paper:

- *EMC Greenplum Data Computing Appliance: Architecture, Performance, and Functions*

Other documentation
For additional information, see the documents and websites listed below:

- *Nagios Exchange* (web site for accessing plug-ins)
- *Nagios* (web site)