

Nagios Certified Professional

Preparation for the Nagios Certified Professional Certification Exam.



Working Lab Manual

This book is designed to be a working manual, a book you can write notes in, underline and use as a reference for a long time. The manual is loaded with Labs to learn and practice skills that you are developing.

Lab – a short training option to illustrate one aspect of the manual

Note: The labs not only contain practical application for information that has already been presented but they also contain new information. This means that the labs are an essential part of the learning process.

Date of Manual Version: March 22, 2012

Copyright and Trademark Information

Nagios is a registered trademark of Nagios Enterprises. Linux is a registered trademark of Linus Torvalds. Ubuntu registered trademarks with Canonical. Windows is a registered trademark of Microsoft Inc. All other brand names and trademarks are properties of their respective owners.

The information contained in this manual represents our best efforts at accuracy, but we do not assume liability or responsibility for any errors that may appear in this manual.

Table of Contents

About This Manual.....	6
Intended Audience.....	6
Preparation for Exercises.....	6
Chapter 1: Introduction.....	1
Nagios Monitoring Solutions.....	1
Technical Support.....	2
Official Training.....	2
Nagios Terminology.....	3
Plugins	3
Host	6
Service	6
Users.....	6
Contacts.....	6
Contactgroups.....	6
Acknowledgment.....	7
Downtime.....	8
Disabled.....	8
Latency.....	9
State.....	9
Host and Service States.....	10
Agents.....	10
Unhandled.....	11
Installation	11
Chapter 2: Configuration	13
Initial Set Up.....	13
Contact Information.....	13
Pre-Flight Check.....	13
Creating a Password.....	15
Eliminating the HTTP Error.....	15
Nagios Check Triangle.....	15
Nagios Checks.....	17
Active.....	17
Passive	18
Security Risks.....	19
Chapter 3: Updates.....	21
Checking for Updates.....	21
Chapter 4: User Management.....	23
Authentication and Privileges.....	23
Authentication.....	23
Notification	28
Multi_Level Notifications.....	31
Escalation.....	34
Notification: Host and Service Dependencies.....	39
Chapter 5: Management.....	41
Web Interface	41

Home.....	41
Documentation.....	42
Tactical Overview.....	43
Map.....	44
Hosts.....	46
Services	49
Host Groups.....	50
Service Groups.....	51
Problems.....	52
Quick Search.....	53
Availability.....	54
Trends.....	56
Alerts.....	58
Notifications.....	62
Event Log.....	62
Comments.....	63
Downtime.....	64
Process Info.....	67
Performance Info	68
Scheduling Queue	69
Configuration.....	70
Event Handlers.....	71
Host Groups.....	74
Service Groups.....	76
Managing Nagios Time.....	77
Nagios Core BackUp.....	78
Reachability	81
Network Outages.....	86
Volatile Service	86
State Stalking.....	86
Flapping.....	86
Resolving Problems.....	89
Disabling Notifications.....	90
Sending Mail From Nagios.....	91
Commit Error from the Web Interface	94
Chapter 6: Monitoring.....	97
Plugin Use.....	97
Monitoring Public Ports.....	97
check_ping.....	98
check_tcp.....	98
check_smtp.....	99
check_imap.....	100
check_simap.....	101
check_ftp.....	101
check_http.....	102
check_mysql.....	104

Monitoring Linux.....	108
NRPE Concepts.....	108
SSH Concepts.....	111
Monitoring Windows.....	113
NSClient++ Concepts.....	113
MSSQL	116
Log Monitoring.....	117
Monitor Nagios Logs.....	118
Network Printers.....	119
Checking Printers with SNMP.....	121
Chapter 7: Practical Exercises.....	125
Exercise #1: Login and Research.....	125
Exercise #2: Responding to Problems.....	130
Exercise #3: Reports.....	135
Exercise #4: Passive vs. Active Checks.....	142

About This Manual

The purpose of this manual is to provide a study resource for the Nagios Certified Professional exam. This manual has been written to aid those taking the exam, but it is also a resource for those who are professionals that will use Nagios on a daily basis, for example those working on a Helpdesk. The questions that are presented in the exam are framed in context in this manual. In order to facilitate learning at a deeper level, exercises are included to help students work through the practical solutions that the exam represents.

Intended Audience

The information contained in this manual is intended for those who will be pursuing the Nagios Certified Professional Certification from Nagios and for professionals working with Nagios on a daily basis. Those taking the exam will find the solutions to the questions on the test within the manual placed in context to help aid the learning process. Often the solutions will be illustrated with screenshots to make it more practical. Those who work at a Helpdesk or those who are in management and need to view the activities on the network and create reports about the network will find this manual helpful as well.

Preparation for Exercises

There are several step-by-step exercises included in the manual which will illustrate these aspects that a professional using Nagios needs to understand:

- * How to handle outages as they occur on the network.
- * How to investigate incidents that occur on the network.
- * How to acknowledge alerts in order to prevent additional notifications and communicate that information to others.
- * How to schedule downtime when hosts or applications need to be intentionally shut down for maintenance.
- * How to generate reports about the network that can be shared.
- * How to predict problems before they occur by analyzing information provided by Nagios.

Generally the exercises can be performed on any network and illustrate skills that all networks using Nagios will employ.

Copyright by Nagios Enterprises, LLC

Cannot be reproduced without written permission. P.O. Box 8154, Saint Paul, MN 55108

Chapter 1: Introduction

Nagios is the industry standard for Open Source network monitoring that provides the ability for an organization to identify and resolve infrastructure problems. Nagios encompasses many features that allow it to accomplish this task. Here is a summary of features:

Flexibility

Flexibility in an ever changing environment is a requirement to modern network monitoring. Nagios has been designed to be able to meet these flexibility requirements by providing the tools to monitor just about anything that is connected to a network. In addition, Nagios allows the administrator to monitor both the internal metrics like CPU, users, disk space, etc. and the application processes on those devices. The flexibility of Nagios Core allows you to use it to perform and schedule checks, perform event handling and alert administrators as needed.

Extensibility

Nagios is designed to be able to use both plugins and addons designed by Nagios as well as be able to implement plugins and addons created by third-party organizations. Nagios is able to integrate with almost any script languages that an organization may be using including; shell scripts, Perl, ruby, etc.

Scalability

As companies grow more equipment will need to be monitored and greater diversity of equipment will be implemented. Nagios is designed to be able to scale with companies as they grow and have changing needs.

Open Source code

Nagios Core is an Open Source Software licensed under the GNU GPL V2.

Customizable

Customization not only includes what devices to monitor, how those devices and applications within the devices will be monitored, but also includes the protocol, plugin, addon, etc, that is incorporated into Nagios to allow that monitoring to occur.

Nagios Monitoring Solutions

Nagios Core is the foundational application that provides the monitoring and alerting options that Nagios is known for. Administration of the Nagios interface is mainly achieved through the CLI or Command Line Interface. The Nagios web interface which uses CGI as the backend by default can be modified to use a MySQL database. The frontend or web interface, can be modified with custom options to provide the look and feel that an organization needs. Several examples of frontends would be themes that are available (i.e. Exfoliation, Vautour and Arana), Web Interfaces like VShell, Nagiosdigger, MNTOS, Check_MK and Mobile Interfaces like Nagios Mobile, NagMobile and iNag. Vshell is the official PHP interface for Nagios Core. Nagios Core by design features and supports many different addons that can be used with it.

Nagios XI takes the Nagios Core and builds upon it to create an enterprise-class monitoring and alerting solution that is easier to set up and configure using a PHP frontend. Nagios XI using easy to use network wizards provides infrastructure monitoring of all of an organizations critical hardware, applications, network devices and network metrics. The dashboard feature allows you to view the entire infrastructure visually as you monitor all of these

services and devices. You also have the alerting options which communicate to administrators when services and hosts have problems. The trending and hardware capacity limits help you create proactive decisions about the network and devices on the network. The graphical interface is easy to customize to fit the organization needs and by monitoring the graphs will help you predict network, hardware and application problems.

Nagios Fusion provides a GUI for central management of a network infrastructure spread over a large geographical area. With central management Nagios Fusion allows the organization to review the organization's entire structure in one location through one interface and yet allow each location to manage their infrastructure independently. Tactical overview screens provide a snapshot of the monitored devices globally.

Nagios Fusion is distributed monitoring the easy way. It provides scalability and comprehensive server support worldwide and in a central location. Fusion also provides the opportunity to create a failover situation with multiple Fusion servers.

Technical Support

The official support site for Nagios can be found at <http://support.nagios.com/forum>. This site provides both free support open to anyone and also customer support for those who have purchase a support contract. The user can ask questions of the technical staff at Nagios and receive answers usually within the same business day.

Official Training

Nagios provides Official Nagios Training for both Nagios Core and Nagios XI. The training options can be found at <http://nagios.com/services/training>. Training services include Live Training performed over the Internet or on-site as well as self-paced training for those wanting to work on their own as they have available time. The Official Nagios training provides users with comprehensive manuals with step-by-step instructions and videos which students can view in order to understand how to implement Nagios in a variety of ways.

Nagios Terminology

Nagios terminology can be a challenge, especially for those who will monitor the Nagios interface but who will not typically install and configure Nagios. This section will try to add clarity to some of the more important terms. If you have problems understanding terms or would like additional information, there is a "Documentation" link under "General" in the menu which may provide answers to questions.

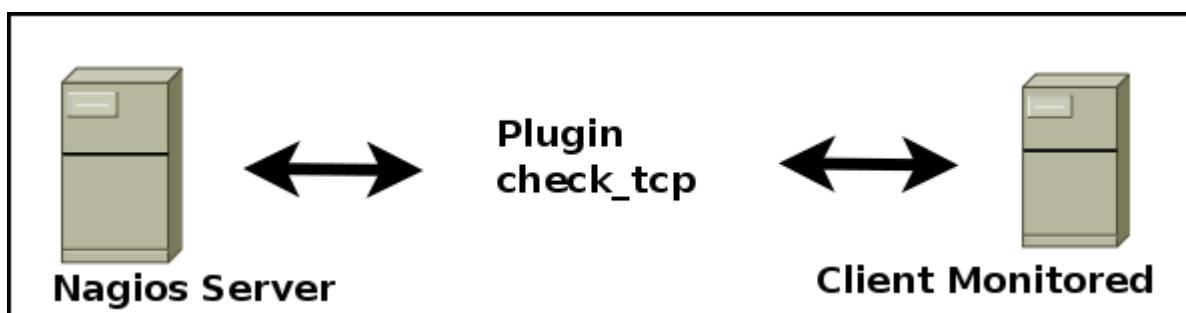
Plugins

Nagios uses plugins which are external programs that can consist of either a script (Perl, shell script, ruby,etc.) or a compiled executable. These plugins are used to check services and hosts on the network. Plugins provide communication between the Nagios core logic process and the hosts and services required to monitor. Each plugin must be configured specifically for the host or service which will be evaluated. Plugins are created separated from the Nagios process so they will need to be downloaded and installed separately. The Official Nagios Plugins are a group of plugins designed, tested and compiled specifically for Nagios. You can download from these locations.

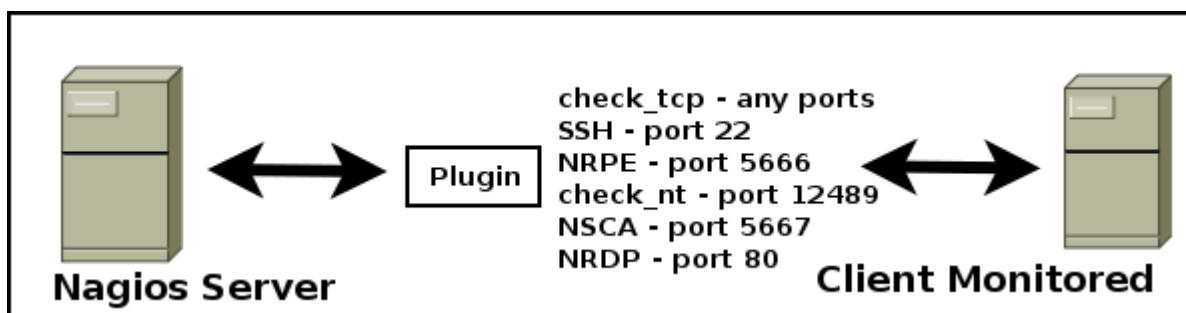
Nagios Plugins

Official Nagios Plugins	http://nagiosplugins.org/
Nagios Plugin Downloads	http://www.nagios.org/download/
NagiosExchange	http://exchange.nagios.org/

Here is an illustration of the Nagios server communicating with the remote client through a plugin.



Plugins can be used to connect to the remote server using various ports and protocols in the communication process.



Here are several examples of how Nagios can connect to a client using plugins.

Public Service Checks

There are a number of protocols that exist which allow the Nagios server to test them externally. For example the common port 80 is available on any web server.

FTP	- port 21
SSH	- port 22
SMTP	- port 25
WEB	- port 80
POP3	- port 110
IMAP	- port 143
Secure Web	- port 443

These public services allow Nagios to not only check to see if the port is open but to verify the correct application is running on the specific port. This can be done because each of these public services run specific protocols which provide the information needed to monitor them correctly and to differentiate them from other services on the same server.

Checks Using SSH

Nagios can connect to a client server using SSH and then execute a local plugin to check internal functions of the server like CPU load, memory, processes, etc. The advantage of using SSH is that checks are secure in the connection and the transfer of information. The disadvantage of SSH is the complexity of setting up keys and the configuration required on the host including editing visudo for some checks.

Nagios Remote Plugin Executor

NRPE, Nagios Remote Plugin Executor, executes plugins internally on the client and then returns that information to the Nagios server. The Nagios server connects on port 5666 in order to execute the internal check. NRPE is protected by the xinetd daemon on the client so that an administrator can restrict the connections to the NRPE plugins. The advantage is that it is the easiest agent to set up.

Monitoring with SNMP

SNMP, Simple Network Management Protocol, is used extensively in network devices, server hardware and software. SNMP is able to monitor just about anything that connects to a network, that is the advantage. The disadvantage is that it is not easy to work with. The complexity of SNMP is made even worse by the fact that vendors write proprietary tools to monitor SNMP that are not easily accessed using Nagios. SNMP can be monitored directly using Nagios plugins or the device itself can monitor SNMP and send information to SNMP traps which can be located on the Nagios server. The difficulties are further aggravated when using traps as the SNMP trap information must be translated into data that Nagios can understand.

Nagios Service Check Acceptor

NSCA, Nagios Service Check Acceptor, employs a daemon on the Nagios server which waits for information generated by passive checks which execute independently on the client being monitored by Nagios. The advantage of NSCA is that services are monitored locally independent of the Nagios server and then sent to the Nagios server so this is a good option when a firewall between the Nagios server and the client prevent other types of communication. The disadvantage is that passive checks use plugins but often require scripts to execute on the client.

Communication can be encrypted between the client and the Nagios server and a password will be required to

complete communication.

Another use for NSCA is distributed monitoring. Distributed monitoring allows a wide geographical base of network devices to be monitored by multiple Nagios servers which use NSCA to send service checks and host checks to a central Nagios server.

Nagios Remote Data Processor

NRDP is another way of monitoring using passive checks. The advantage of using NRDP is that it uses less resources and it connects on the common port 80 or 443 on the Nagios server.

NSClient ++

This agent is installed on Windows servers and desktops in order to monitor with either check_nt (port 12489), NRPE (port 5666) or using passive checks. This is the most reliable Windows agent available and has the advantage of multiple options for monitoring.

Currently the plugins provided in the nagios-plugins package provides about 80 plugins and another 80 in the contrib directory. This certainly provides you with adequate plugins to get started. By searching Google, SourceForge and GitHub you will be able to find additional plugins.

If you need to find out more information about a specific plugin you can use this command after moving into the plugins directory:

```
./<plugin_name> --help
```

The “help” feature provides the version, structure and options that the plugin uses. Often examples of how to use the plugin are included as well.

Here is an example of the common ping plugin.

```
./check_ping --help
check_ping v1.4.15 (nagios-plugins 1.4.15)
Copyright (c) 1999 Ethan Galstad <nagios@nagios.org>
Copyright (c) 2000-2007 Nagios Plugin Development Team
    <nagiosplug-devel@lists.sourceforge.net>
```

Use ping to check connection statistics for a remote host.

Usage:
check_ping -H <host_address> -w <wrta>,<wpl>% -c <crta>,<cpl>%
 [-p packets] [-t timeout] [-4|-6]

Options:
-h, --help
 Print detailed help screen
-V, --version
 Print version information
-4, --use-ipv4
 Use IPv4 connection
-6, --use-ipv6
 Use IPv6 connection

```
-H, --hostname=HOST
    host to ping
-w, --warning=THRESHOLD
    warning threshold pair
-c, --critical=THRESHOLD
    critical threshold pair
-p, --packets=INTEGER
    number of ICMP ECHO packets to send (Default: 5)
-L, --link
    show HTML in the plugin output (obsoleted by urlize)
-t, --timeout=INTEGER
    Seconds before connection times out (default: 10)
```

Host

A host is a server, switch, router, printer or any other network device that you want to monitor. Nagios requires an IP Address for the host or a FQDN (Fully Qualified Domain Name) to determine the exact location of the device. Each host must also have a unique name that will tie the host name reference to the IP Address or FQDN. The host information is required to be able to configure a check for the host or service.

Service

A service is any metric that may be required to evaluate on the host, including internal metrics like CPU, memory, users, disk space, etc. A service can also monitor daemons and applications running on the server like a database, MySQL or Postfix for example.

Users

Users is a reference to an individual that has been given access to the Nagios web interface in order to view hosts and services and in order to manage those hosts and services. Note: users and contacts are different as users access the web interface and contacts receive notifications. However, a user can be set up to also be a contact.

Contacts

Contacts are the individual administrators that are notified by Nagios because of a host or service problem. These contacts are typically a part of a contactgroup. The contact information provides a way to communicate to the administrator.

Contactgroups

These groups are the connection between detected problems and communication with individuals in the group. Contactgroups usually are related to the type of device. For example, organizations may group windows administrators into a separate group from Linux administrators as often the skills required to solve problems is quite different. Contactgroups provide an excellent way to manage notification in a rapidly changing environment.

Acknowledgment

Acknowledgment will temporarily suppress alert notifications until the host or service returns to an OK state. This can be achieved inside the web interface by selecting the service and then choosing “Acknowledge this service problem” (see image).

Service Information Last Updated: Fri Mar 16 06:24:41 MDT 2012 Updated every 90 seconds Nagios® Core™ 3.3.1 - www.nagios.org Logged in as nagiosadmin <ul style="list-style-type: none"> View Information For This Host View Status Detail For This Host View Alert History For This Service View Trends For This Service View Alert Histogram For This Service View Availability Report For This Service View Notifications For This Service 	Service AIDE On Host Company Server (bash) Member of No servicegroups. 192.168.5.190
Service State Information	
Current Status: CRITICAL (for 42d 15h 9m 38s) Status Information: <pre>added: /etc/rc.d/rc3.d/S65dovecot added: /etc/httpd/sales changed: /etc/profile.d changed: /etc/profile.d/krb5-devel.sh changed: /etc/profile.d/krb5-devel.csh changed: /etc/php.ini changed: /etc/postfix changed: /etc/postfix/main.cf changed: /etc/hosts changed: /etc/prelink.cache changed: /etc/rc.d/rc1.d changed: /etc/rc.d/rc1.d/S26lvm2-monitor changed: /etc/rc.d/rc1.d/K35dovecot changed: /etc/rc.d/init.d changed: /etc/rc.d/init.d/lvm2-monitor</pre>	Service Commands <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Disable active checks of this service <input type="radio"/> Re-schedule the next check of this service <input type="radio"/> Submit passive check result for this service <input checked="" type="checkbox"/> Stop accepting passive checks for this service <input checked="" type="checkbox"/> Stop obsessing over this service <input checked="" type="checkbox"/> Acknowledge this service problem <input checked="" type="checkbox"/> Disable notifications for this service <input type="radio"/> Delay next service notification <input type="checkbox"/> Send custom service notification <input type="radio"/> Schedule downtime for this service <input checked="" type="checkbox"/> Disable event handler for this service <input checked="" type="checkbox"/> Disable flap detection for this service

Once that option is selected the administrator can enter the reason for the problem and that it is currently the issue. Enter the host name, the service, your name and the comment you want to communicate to other administrators (these are all required). You also can select if the comment will be sticky ,persistent or if you want to send notification. The “Sticky Acknowledgement”, when it is checked, will prevent further notifications if the problem continues. “Persistent Comment” in Nagios 3 will retain the comment even after a reboot and must be manually unchecked when it is fixed. If you leave it unchecked Nagios will remove the comment when a solution is found.

External Command Interface
 Last Updated: Fri Mar 16 06:28:27 MDT 2012
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

You are requesting to acknowledge a service problem

Command Options

Host Name:	bash
Service:	AIDE
Sticky Acknowledgement:	<input checked="" type="checkbox"/>
Send Notification:	<input checked="" type="checkbox"/>
Persistent Comment:	<input type="checkbox"/>
Author (Your Name):	Nagios Admin
Comment:	I am currently updating several programs.
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

Other administrators may now see that the problem is being worked on by viewing System and Comments in the menu.

Service Comments

↻ [Add a new service comment](#)

Host Name	Service	Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
bash	AIDE	03-16-2012 06:31:02	Nagios Admin	I am currently updating several programs.	214	No	Acknowledgement	N/A	

If the host or service was disabled permanently then a better solution is to disable checks permanently.

Downtime

If you are going to work on a server or device and need to schedule downtime so Nagios does not notify administrators that can be performed at the web interface. When you select the host or service that will be down you have an option to schedule downtime. When downtime is scheduled Nagios will place a comment in the web interface in order to communicate the fact to all administrators who access the web interface.

Disabled

“Disabled” is a term which refers to turning off a feature that Nagios provides. For example, active checks, passive checks, obsessing, notifications, event handlers or flap detection. The example shows the option on the right menu was selected to “Disable flap detection for this service”.

Service Information

Last Updated: Mon Mar 19 15:08:46 MDT 2012
Updated every 90 seconds
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as [nagiosadmin](#)

[View Information For This Host](#)
[View Status Detail For This Host](#)
[View Alert History For This Service](#)
[View Trends For This Service](#)
[View Alert Histogram For This Service](#)
[View Availability Report For This Service](#)
[View Notifications For This Service](#)

Service
Nagios DNS
On Host
Company Server
(bash)

Member of
No servicegroups.

192.168.5.190

Service State Information

Current Status:	OK (for 27d 11h 47m 27s)
Status Information:	DNS OK: 0.108 seconds response time. nagios.com returns 173.45.234.224
Performance Data:	time=0.108438s;;0.000000
Current Attempt:	1/3 (HARD state)
Last Check Time:	03-19-2012 15:06:36
Check Type:	ACTIVE
Check Latency / Duration:	0.237 / 0.154 seconds
Next Scheduled Check:	03-19-2012 15:16:36
Last State Change:	02-21-2012 02:21:19
Last Notification:	N/A (notification 0)
Is This Service Flapping?	N/A
In Scheduled Downtime?	NO
Last Update:	03-19-2012 15:08:42 (0d 0h 0m 4s ago)

Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	DISABLED

Service Commands

- X** Disable active checks of this service
- C** Re-schedule the next check of this service
- ?** Submit passive check result for this service
- X** Stop accepting passive checks for this service
- X** Stop obsessing over this service
- X** Disable notifications for this service
- C** Send custom service notification
- L** Schedule downtime for this service
- X** Disable event handler for this service
- V** Enable flap detection for this service

Latency

Latency is the difference between when a check is scheduled to run and when it does actually run. Latency is often used as a metric for Nagios performance. The greater the time difference between when a check was scheduled to run and when it actually runs means a greater degradation in performance. Latency can be observed by choosing the menu on the web interface and selecting “Performance Info” under System. The latency for service and host checks is listed. See Performance Info under the Web Interface for how to view this information.

State

Nagios has a built in protection mechanism against false positives called state. State is measured by two values SOFT and HARD. When a failure is detected for a host or service which was originally working, the initial state is SOFT, which does not create a notification. Nagios checks several times before a state is determined to be HARD, which does create a notification. The max_check_attempts setting is used to determine how many times Nagios rechecks a SOFT state before it is moved to a HARD state. So if the max_check_attempts setting is 5, Nagios will check 5 times and remain in a SOFT state, until another check is made to create a HARD state. Over this time, the SOFT state must remain in a non-OK state for it to be moved to a HARD state. It is the HARD state which triggers notification.

In the example, the HARD state is illustrated in this service check, which shows the max_check_attempts setting is 3 and the first of 3 checks has occurred.

Service State Information	
Current Status:	OK (for 13d 8h 25m 53s)
Status Information:	DISK OK - free space: / 1155 MB (38% inode=79%): /=1869MB;2419;2721;0;3024
Performance Data:	
Current Attempt:	1/3 (HARD state)
Last Check Time:	03-20-2012 02:51:45
Check Type:	ACTIVE
Check Latency / Duration:	0.077 / 0.033 seconds
Next Scheduled Check:	03-20-2012 03:01:45
Last State Change:	03-06-2012 17:31:44
Last Notification:	N/A (notification 0)
Is This Service Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	03-20-2012 02:57:31 (0d 0h 0m 6s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host and Service States

Nagios uses four different return values from plugins in order to determine the state of a host or service. These four values determine the color of the output in the web interface.

0	OK	green
1	WARNING	yellow
2	CRITICAL	red
3	UNKNOWN	orange

The color changes are what is reflected in the web interface.

Agents

Agents refer to the programs, usually daemons, that must be placed on the client to listen for connections coming from the Nagios server.

Typical Agents

SSH	- port 22
NRPE	- port 5666
NSClient++	- ports 5666 or 12489

Agents provide greater access to the clients internal metrics as plugins can be run on the client by Nagios using the agent that is installed on the client.

Unhandled

Unhandled host or services are those in non-OK states which have not been acknowledged, are not in a scheduled downtime and if they are services they are associated with a host that is not in a problem state.

Installation

Understanding installation options is an important part of troubleshooting as the installation method determines the location of binaries, configuration files and plugins. These locations may even differ based on the version of the Linux distribution. The following chart provides common locations using the examples of compiling, using a CentOS RPM or using a Debian/Ubuntu Deb file. The point is, know how Nagios was installed before starting the troubleshooting process.

NAGIOS	Program Location	Configuration File	Plugins
Compile	/usr/local/nagios/bin/nagios	/usr/local/nagios/etc/nagios.cfg	/usr/local/nagios/libexec
CentOS	/usr/bin/nagios	/etc/nagios/nagios.cfg	/usr/lib/nagios/plugins
Debian/Ubuntu	/usr/bin/nagios3	/etc/nagios3/nagios.cfg	/usr/lib/nagios/plugins
Web Server	Program Location	Web Server Configuration	Nagios Web Config
CentOS	/usr/sbin/httpd	/etc/httpd/conf/httpd.conf	/etc/httpd/conf.d/nagios.cfg
Debian/Ubuntu	/usr/sbin/apache2	/etc/apache2/apache2.conf	/etc/nagios3/apache2.conf
Users	htpasswd Database		
Compile	/usr/local/nagios/etc		
CentOS	/etc/nagios		
Debian/Ubuntu	/etc/nagios3/		

The implications for documentation are that you must translate any documentation to the installation method that was chosen.

Installation from source is a process where the source code that was developed by the programmer is converted into a binary format that the server can run. Compiling Nagios may require a few extra steps in setting up Nagios but there are several advantages over using a RPM repository or a DEB repository. The biggest advantage of installing from source is that the installation process can be repeated on almost any Linux distribution and therefore each distribution will have the same location for binaries, configuration files and plugins.

Chapter 2: Configuration

Nagios configuration can be a complex process over a long period of time. Gaining a basic outline of that process can be useful in troubleshooting and determining the intricacies of how Nagios functions.

Initial Set Up

Whether you are installing using a repository or from source, there are some initial steps to take to get started. The first step is to add a contact email for the nagiosadmin. The user nagiosadmin by default is the only user able to access the whole web interface.

Contact Information

In order to receive notification of a problem, the nagiosadmin user must have a valid email address configured. Edit `/usr/local/nagios/etc/objects/contacts.cfg` (RPM repository `/etc/nagios/objects/contacts.cfg`). Place nagiosadmin user email in the email location.

```
define contact{
    contact_name          nagiosadmin      ; Short name of user
    use                   generic-contact   ; Inherit default values
    alias                Nagios Admin     ; Full name of user
    email                your_email      ; <<***** CHANGE THIS TO YOUR EMAIL
}
```

Restart Nagios to make the changes take effect.

Pre-Flight Check

The pre-flight check is a procedure that checks the configuration of Nagios and returns any errors, before Nagios is started. This becomes a necessary process before restarting Nagios in order to maintain the integrity of the system. Nagios will restart when it encounters Warnings but will not restart if it encounters Errors. In order to use the pre-flight check execute the Nagios binary and point it to the location of the nagios.cfg file, using the verbose option “-v”.

```
nagios -v /usr/local/nagios/etc/nagios.cfg
(RPM repository /etc/nagios/nagios.cfg)
```

```
Nagios 3.3.1
Copyright (c) 1999-2008 Ethan Galstad (http://www.nagios.org)
Last Modified: 12-01-2008
License: GPL
```

```
Reading configuration data...
```

```
Running pre-flight check on configuration data...
```

```
Checking services...
    Checked 8 services.
Checking hosts...
    Checked 1 hosts.
Checking host groups...
    Checked 1 host groups.
Checking service groups...
    Checked 0 service groups.
Checking contacts...
    Checked 1 contacts.
Checking contact groups...
    Checked 1 contact groups.
Checking service escalations...
    Checked 0 service escalations.
Checking service dependencies...
    Checked 0 service dependencies.
Checking host escalations...
    Checked 0 host escalations.
Checking host dependencies...
    Checked 0 host dependencies.
Checking commands...
    Checked 24 commands.
Checking time periods...
    Checked 5 time periods.
Checking for circular paths between hosts...
Checking for circular host and service dependencies...
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...
```

```
Total Warnings: 0
Total Errors: 0
```

Things look okay - No serious problems were detected during the pre-flight check

The output of the pre-flight check, as in this example, clearly indicates everything is OK and the changes that were made can be applied to Nagios safely.

Creating a Password

The nagiosadmin user is created by default and will allow access to the web interface using that pre-created username. However, the password for that user should be created. Execute the following command in order to create a new user/password combination.

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Now login to the web interface with `http://ip_address/nagios`.

Note: The “-c” option in the command above creates a new file, erasing an old file if it exists. Therefore, if the `htpassword.users` file has been created, use the command without the “-c” to create additional users/passwords.

Eliminating the HTTP Error

When you set up the Nagios server and either review your log files in `/var/log/nagios/nagios.log` or review the web interface you may initially see an error related to the web server. The error is related to the fact that you do not have a `index.html` file that exists. **Note: If you do not see the error it is because you have the necessary files so you can skip this step.** Here is what it will look like in the log.

```
WARNING: HTTP/1.1 403 Forbidden - 5240 bytes in 0.001 second response time
Sep 26 10:00:18 nagios nagios: SERVICE ALERT: localhost;HTTP;WARNING;HARD;4;HTTP
```

You can easily eliminate the error by creating an `index.html` file. Create a simple HTML.

```
vi /var/www/html/index.html

<HTML>
<BODY>
Nagios Server
</BODY>
</HTML>

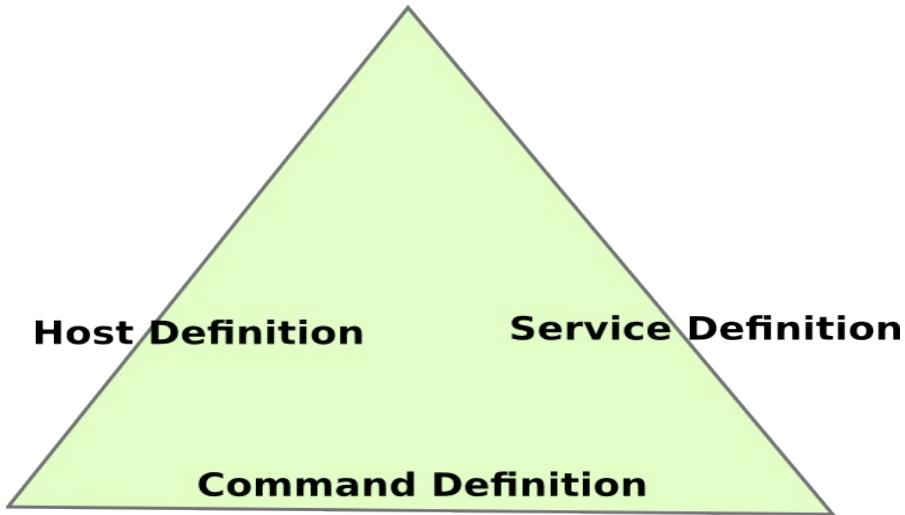
chmod 755 /var/www/html/index.html
chown apache:apache /var/www/html/index.html
```

The example is for a CentOS system. If you are using Debian or Ubuntu for example the name of the web server and the location of the web directory are different.

Nagios Check Triangle

One of the major concepts of creating checks is to remember that all plugins with Nagios will require three elements to be configured. There must be a host definition, a service definition and a command definition. Think of it as a triangle each time you want to use a plugin.

Nagios Check Triangle



These three definitions are all located in three separate files, hosts.cfg, services.cfg and commands.cfg. The name of the file can be different. You may need to create hosts.cfg and services.cfg as they are not created by default. These files are typically located in this directory if you compiled:

/usr/local/nagios/etc/objects

Host Defintion

Nagios needs to know an IP Address of the host you want to check. This is configured in the hosts.cfg file. The hosts.cfg file does not exist initially so you will need to create it. In this example the host_name is “win2008” and it is tied to the address “192.168.3.114”. This is the information Nagios must have to know where to point a request and how to record information for a specific host.

```
define host{
    use windows-server
    host_name win2008
    alias Windows Server
    address 192.168.3.114
}
```

Service Definition

The second part of the triangle is the service definition. Nagios needs to know what service you want to check, so that service or plugin must be defined. In this example the host “win2008”, which Nagios knows now is tied to the IP Address 192.168.3.114, is being checked with the ping plugin. So you can see the host_name determines which host the plugin acts upon and then the service_description is really the text that shows up in the web interface. The check_command, defines the parameters of the plugin. Here you can see that “check_ping” is the plugin and it is followed by two different sections of options divided by “!”. The first section, “60.0,5%”, provides a warning level if

packets are take longer than 60 milliseconds or if there is greater than a 5% loss of packets when the ping command is performed. The second section is the critical level where a CRITICAL state will be created if packets take longer than 100 milliseconds or if there is more than 10% packet loss.

```
define service{
    use                         generic-service
    host_name                   win2008
    service_description          Ping
    check_command                check_ping!60.0,5%!100.0,10%
}
}
```

Command Definition

The command definitions are located in the commands.cfg file which is created by default in the objects directory. Many commands are already defined so you do not have to do anything with those. The check_ping command is one example that has been defined. The command_name, “check_ping”, is what is part of the service definition. The command_line specifically defines where the plugin is located with the “\$USER1\$” macro. This is equal to saying that the plugin check_ping is located in /usr/local/nagios/libexec (if you compiled). The other 4 options include the host, using the \$HOSTADDRESS\$ macro, a warning level (-w) using the \$ARG1\$ macro, the critical level (-c) using the \$ARG2\$ macro and the number of pings to use by default (-p 5).

```
# 'check_ping' command definition
define command{
    command_name    check_ping
    command_line    $USER1$/check_ping -H $HOSTADDRESS$ -w $ARG1$ -c $ARG2$ -p 5
}
```

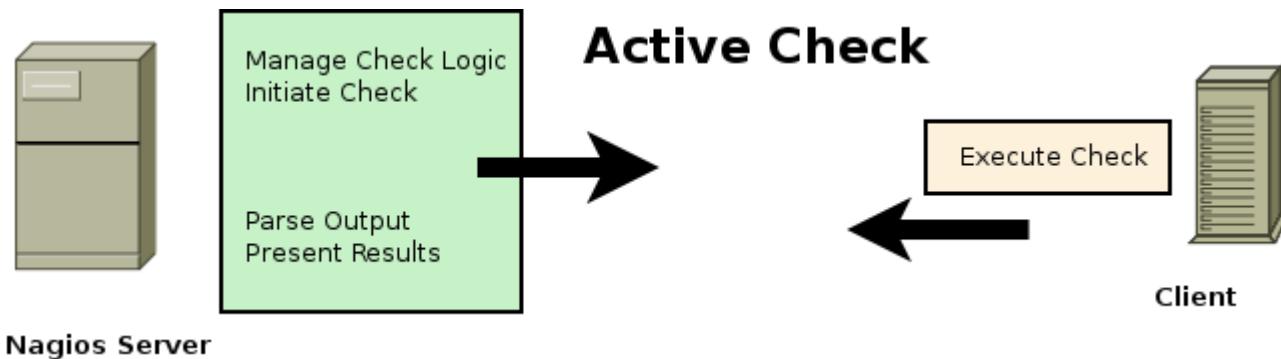
In each of the elements of the Nagios triangle you can see the importance of the term “definition” as each element must be clearly defined and each element is dependent upon the other definitions.

Nagios Checks

Nagios can perform checks two different ways; active or passive. Understanding which method is being used is key to troubleshooting. When comparing active and passive checks one of the biggest differences is that in active checks Nagios explicitly controls each step but in passive checks Nagios is at the mercy of the external host sending data to be processed. In passive checks the client performs the check itself and provides the information to the Nagios server at the interval determined by the client, not Nagios.

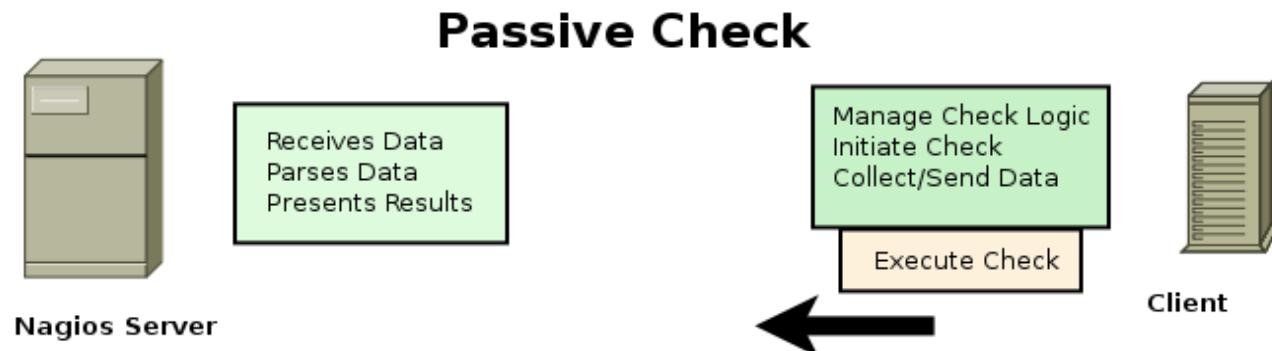
Active

With active checks Nagios initiates and manages each step of the process. This means each step of the process is closely monitored and manipulated by Nagios. The schedule for when checks occur, the organization of resources and the initialization of those resources are controlled by Nagios. The scheduling queue is an example. During time of heavy load Nagios may push this schedule to control activity.



Passive

Typically passive checks are used when a firewall prevents the Nagios server to make a request to the client or when the client is running an application that asynchronous, in other words the time schedule for a service is erratic and cannot be fully determined. Security events is one example of a situation where you do not know when the event may occur. Passive checks may also be used for distributed monitoring where you have multiple Nagios servers providing information to a master Nagios server.



When Passive Checks are used the client uses a program called NSCA (Nagios Service Check Acceptor) and the evaluation occurs locally on the client and then is sent to the Nagios server using NSCA. NSCA runs on the Nagios server as a daemon protected by xinetd. The daemon will listen for requests on port 5667 sent by the client. When the server receives the request the remote server is authenticated using a password that is shared between the Nagios server and the client. The password is encrypted on one of 22 levels to protect it as it moves over the network.

A similar passive check can be used with NRDP (Nagios Remote Data Processor) which communicates to the Nagios server using a secret token and connects on port 80 or 443.

The Nagios server only processes passive checks that are sent to it. In other words, the client must automate the checks using a cron job or the passive checks must be a response to an event.

Security Risks

Nagios can pose security risks to organizations which do not configure Nagios properly. Because the Nagios server is able to execute commands on the hosts that it monitors, special care should go into protecting the Nagios server. Here are a few of the items that need to be considered:

- * use a firewall on the Nagios server to limit access to administrators and client machines that will be sending passive check data
- * encrypt communication to protect data that will be over a public network
- * restrict user privileges to only what is required to administer the Nagios server
- * use a box dedicated to Nagios
- * monitor changes on the Nagios server
- * tighten security on the clients that Nagios will monitor

Chapter 3: Updates

Keeping your Nagios installation up to date is an important part of administration. However, this should only be performed when you have an established backup and restore process just as in any situation.

Checking for Updates

The web interface for Core allows you to easily check for updates by going to the home page.

The screenshot shows the Nagios Core web interface. On the left is a vertical navigation menu with sections: General (Home, Documentation), Current Status (Tactical Overview, Map, Hosts, Services, Host Groups, Service Groups, Problems), Reports (Availability, Trends, Alerts, History, Summary, Histogram), Notifications, Event Log, System (Comments, Downtime, Process Info). The main content area features the Nagios Core logo and version information (Version 3.3.1, July 25, 2011). It includes a "Check for updates" link. Below this are four boxes: "Get Started" (monitoring infrastructure, changing look, addons, support, training, certification), "Don't Miss..." (Nagios World Conference North America, September 27-29, 2011), "Quick Links" (Nagios Library, Labs, Exchange), and "Latest News" (conference sold out, hiring engineer). At the bottom, there's a copyright notice (2010-2011 Nagios Core Development Team and Community Contributors), a license statement (GNU General Public License), and logos for Nagios and SourceForge.NET.

This page also contains links for training, certification, tutorials, labs, plugins and provides the latest Nagios news. By selecting the “Check for updates” the link will assess the current version to see if it is up to date.



The Nagios logo features the word "Nagios" in a bold, black, sans-serif font. A registered trademark symbol (®) is positioned at the top right of the letter "o".

Home News Products Documentation Support Projects About Get Involved What's New Download Conference

Home | Nagios Update Check

Nagios Update Check

| Print | E-mail

Up To Date

Your installation of Nagios Core (3.3.1) is up-to-date, so no upgrade is required. The latest version of Nagios Core is 3.3.1 was released on 2011-07-25.

 SHARE

Search

Contact Us

Phone: 1-888-NAGIOS-1
Email: sales@nagios.com

Stay Informed

Subscribe To Our Newsletter
Privacy by  SafeSubscribeSM

If your version of Nagios Core is out of date it is important that the first thing you do is backup the current version before proceeding to the update process.

Chapter 4: User Management

Users and contacts can be separate functions within Nagios. Users are individual accounts that have access to the web interface. Contacts are users who will be sent notification if there are problems with hosts or services.

Authentication and Privileges

The authentication parameters in the cgi.cfg is a way to configure access so that the contacts that log in must match the hosts and services which they are responsible for. This eliminates them being able to access other hosts.

In order to provide access for a user to be able to see all computers and services on the Web Interface you will need to activate these two parameters on the /usr/local/nagios/etc/cgi.cfg (RPM repository /etc/nagios/cgi.cfg).

```
authorized_for_all_services=fred  
authorized_for_all_hosts=fred
```

If you want to allow a user (like fred) to run any commands on the web interface even if they are not listed with permissions that match a service or host you will need to modify these two parameters.

```
authorized_for_all_service_commands=nagiosadmin,fred  
authorized_for_all_host_commands=nagiosadmin,fred
```

If you wanted to set up configuration so that all users who authenticate to the web interface can do everything they choose, not recommended, then you would place a “*” at the end of each line for all users.

```
authorized_for_all_services=*  
authorized_for_all_hosts=*  
authorized_for_all_service_commands=*  
authorized_for_all_host_commands=*
```

Authentication

Authentication is the process that allows users to access the web interface. Authentication is controlled by the use of a database using the htpasswd command. The database, called htpasswd.users, is located in the /usr/local/nagios/etc directory (/etc/nagios if using the RPM repository). The name and location of the database is determined by the configuration options found in /etc/httpd/conf.d/nagios.conf. In this example, from a CentOS install, you can see that several directories require authentication from this database.

```
ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"
```

```
<Directory "/usr/local/nagios/sbin">
# SSLRequireSSL
Options ExecCGI
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
</Directory>

Alias /nagios "/usr/local/nagios/share"

<Directory "/usr/local/nagios/share">
# SSLRequireSSL
Options None
AllowOverride None
Order allow,deny
Allow from all
# Order deny,allow
# Deny from all
# Allow from 127.0.0.1
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
Require valid-user
</Directory>
```

Access is maintained through the database but the permissions a user has once they authenticate are determined by contacts, contact groups and cgi permissions determined from the cgi.cfg file. An important point to remember when setting up permissions is that the contact is only able to see the host or service that they are responsible for by default. Make sure contact names match the user created for access to the web interface.

These settings represent the default settings in the cgi.cfg file for permissions to the web interface. The user “nagiosadmin” is the default nagios user with access and unlimited permissions to the web interface. The defaults demonstrate why it is so important to correctly set up the nagiosadmin user as part of the initial configuration.

```
use_authentication=1
use_ssl_authentication=0
#default_user_name=guest
authorized_for_system_information=nagiosadmin
authorized_for_configuration_information=nagiosadmin
authorized_for_system_commands=nagiosadmin
authorized_for_all_services=nagiosadmin
authorized_for_all_hosts=nagiosadmin
authorized_for_all_service_commands=nagiosadmin
```

```
authorized_for_all_host_commands=nagiosadmin  
#authorized_for_read_only=user1,user2
```

Scenario: Turn Off All Authentication

Turning off all authentication is not recommended under any circumstances. It is only demonstrated here in order to aid in the understanding of how Nagios authentication works. These changes allow anyone to make changes to the Nagios interface, hosts and services.



Security Tip

Warning, this is a serious security issue and should not be implemented.

There are two steps required to turn off all security. Edit the cgi.cfg file located in /usr/local/nagios/etc (/etc/nagios if using the RPM repository) and change the “use_authentication” to a “0”.

```
use_authentication=0
```

The second step required is to access the /etc/httpd/conf.d/nagios.conf file and comment out the lines that require authentication for the Nagios directories.

```
ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"  
  
<Directory "/usr/local/nagios/sbin">  
# SSLRequireSSL  
Options ExecCGI  
AllowOverride None  
Order allow,deny  
Allow from all  
# Order deny,allow  
# Deny from all  
# Allow from 127.0.0.1  
# AuthName "Nagios Access"  
# AuthType Basic  
# AuthUserFile /usr/local/nagios/etc/htpasswd.users  
# Require valid-user  
</Directory>  
  
Alias /nagios "/usr/local/nagios/share"  
  
<Directory "/usr/local/nagios/share">  
# SSLRequireSSL  
Options None  
AllowOverride None  
Order allow,deny  
Allow from all  
# Order deny,allow  
# Deny from all  
# Allow from 127.0.0.1  
# AuthName "Nagios Access"
```

```
# AuthType Basic
# AuthUserFile /usr/local/nagios/etc/htpasswd.users
# Require valid-user
</Directory>
```

Restart Nagios and the web server.

Scenario: Create a View Only Account

This scenario will create a user that can view all hosts and services but not be allowed to make any changes to those hosts or services. This is typically the settings you may choose for management to review the status of hosts and services.

Create the user in the htpasswd.users database.

```
htpasswd htpasswd.users management
New password:
Re-type new password:
```

Make modifications to the cgi.cfg file by adding the user separated by a comma, without spaces. The user has global access, which means they are not required to be listed as contacts for hosts and services. The user is also added to the read only list.

```
authorized_for_all_services=nagiosadmin,management
authorized_for_all_hosts=nagiosadmin,management
authorized_for_read_only=management
```

Restart Nagios and the web server.

Scenario: Create System Administrator with No Contact Information

In this scenario the settings will allow a user to have full access to all settings on all hosts and services just like the nagiosadmin user. However, this user is not associated with any contact information so will not be notified at any time. This account is strictly administration only.

```
htpasswd htpasswd.users john
New password:
Re-type new password:
```

Edit the cgi.cfg file and add john to each of the lists indicated below.

```
authorized_for_system_information=nagiosadmin,john
authorized_for_configuration_information=nagiosadm,john
authorized_for_system_commands=nagiosadmin,john
authorized_for_all_services=nagiosadmin,john
authorized_for_all_hosts=nagiosadmin,john
authorized_for_all_service_commands=nagiosadmin,john
authorized_for_all_host_commands=nagiosadmin,john
```

Restart Nagios and the web server.

Scenario: Create an Administrator with Limited Access

This user will only be allowed to access the hosts and services that they are associated with via contact information. This may be the type of settings used when an organization has divided responsibilities for routers, Windows servers and Linux servers for example.

```
htpasswd htpasswd.users sue
New password:
Re-type new password:
```

Create a new contact entry in contacts.cfg and specify the contact_name, alias and email contact information for the user.

```
define contact{
    contact_name          sue
    use                   generic-contact
    alias                Router Admin
    email                sue@example.com
}
```

Add the user to a group or create a new group in the contacts.cfg file. This example shows a user added to a new contact group called router-admins. By creating a new group it enables an administrator to assign that group to a series of devices, like routers.

```
define contactgroup{
    contactgroup_name     router-admins
    alias                Router Administrators
    members              sue
}
```

At this point you will need to edit the hosts and services and add the “contact_groups router-admins” which will override the default settings in the template. This will enable only those users in this contact group access to these hosts and services unless they have global access from the cgi.cfg file.

```
define host{
    use                  generic-switch
    host_name           cisco
    alias               cisco router
    address             192.168.5.220
    contact_groups      router-admins
}
define service{
    use                  generic-service
    host_name           cisco
    service_description PING
    check_command       check_ping!200.0,20%!600.0,60%
    normal_check_interval 5
    retry_check_interval 1
}
```

```
    contact_groups          router-admins
}
```

Restart Nagios and the web server.

Notification

Nagios provides the ability to notify administrators when problems develop. The key to working with Nagios on notifications is to avoid the false alarms that can be very frustrating to those who have to come in and fix the reported problems.

Notification is triggered when the max_check_attempts parameter has been reached and a HARD state has occurred. In other words, it is confirmed that the machine has moved from a functioning state to a broken state. So here is also a key to preventing false alarms, move the max_check_attempts for a service or host to a higher number so that it must check a number of times before it notifies administrators. Here is an example that requires 10 attempts.

```
max_check_attempts      10
```

Until the max_check_attempts has been reached, Nagios still considers this a SOFT state. The other important point here is that these 10 attempts must all return a CRITICAL status consecutively before a HARD state is reached. In other words, if you change to 10 as the max_check_attempts, the hard state is reached when it returns 10 CRITICAL states in a row.

The notification process flows through a number of filtering options that you can provide for a fine tuned set up.

System Wide Filtering

Notifications can be turned on or off by editing the nagios.cfg file, “1” indicating that it is on and “0” indicating it should be off system wide. The default is to have it on.

```
enable_notifications=1
```

This setting will automatically take into account downtime which is scheduled in the web interface.

Service / Host Filtering

Notifications are sent for host objects for d(down), u(unreachable), r(recovered), f(flapping-up then down state) and now with Nagios 3 s(start of planned maintenance). Notifications are sent for service objects for c(critical), w(warning), u(unknown), r(recovered), f(flapping) and s(start planned maintenance). These options can be entered into the options line to select those options that you want notifications for.

```
notification_options=c,r
```

If you set notifications to n(None) or "0" it will not send notifications. Each host and service references a template, “use generic-switch”. The specific settings you enter into a host or service definition will override the template settings. In this example, this host will not send notifications regardless of the settings which are in the template and the global settings as well.

```
define host{
```

```
use          generic-switch
host_name    zyxel
alias        zyxel router
notifications_enabled 0
address      192.168.5.79
}
```

Service Groups

You will need to create a file called servicegroups.cfg and put an entry in nagios.cfg to indicate where it is. Note the entries are in pairs (first host, then service) “web,HTTP”.

```
# SERVICE GROUPS
define servicegroup{
    servicegroup_name      webservice
    alias                  Web-Sites
    members                web,HTTP ,web2,HTTP
}
}
```

Host Groups

Create an entry in nagios.cfg to the location of hostgroups.cfg.

```
define hostgroup{
    hostgroup_name         linux-web
    alias                  Linux-Sites
    hostgroup_members      web
}
}
```

The notification_period parameter allows you to set when these notifications should occur. Here are several examples of timeperiods. Note that time parameters must be defined in the timeperiods.cfg so that these can be used.

```
notification_period   24x7h
notification_period   workhours
notification_period   shift2
notification_period   shift3
```

The notification_interval is an parameter that you can adjust so that you will be able to determine how often you will be notified of a situation. The default setting is in minutes. This example will notify the contacts every 60 minutes.

```
notification_interval 60
```

If you only wanted one notification sent you can set this parameter to “0”.

```
define host{
    name                  linux-box
    use                   generic-host
    check_period          24x7
    check_interval        5
    retry_interval        1
    max_check_attempts   10
    check_command         check-host-alive
    notification_period  24x7
    notification_interval 0
    contact_groups       admins
    register              0
}
```

Contact Filtering

Nagios will allow you to determine who gets contacted for each situation. It also allows you to create groups so that different administrators can be contacted for different reasons and at different times.

Contact Admins

```
define contact{
    contact_name          nagiosadmin
    use                   generic-contact
    alias                Nagios Admin
    email                jane@some_email.com
}

define contact{
    contact_name          joe
    alias                Win Admin
    use                   generic-contact
    service_notification_period workhours
    host_notification_period workhours
    service_notification_options w,u,c,r
    host_notification_options d,r
    service_notification_commands notify-service-by-email
    host_notification_commands notify-host-by-email
    email                joe@some_email.com
}
```

Contact Groups

Notice that each group has a contact name and the members are different. An important aspect of contacting different admins at different times is that you want to consistently run the service and host to continue sending messages so that you do not miss contact with an administrator. So the result is that the service and host send messages 24x7 and on a regular interval but the admins are divided by who accepts responsibility during different time periods and what groups they are in.

```
define contactgroup{
    contactgroup_name     admins
    alias                Nagios Administrators
```

```
members          nagiosadmin
}
define contactgroup{
    contactgroup_name      win-admins
    alias                  Windows Administrators
    members                joe
}
```

Here is an example of the many notification options that you have. Notice that the choices are spread quite wide over the whole spectrum of Nagios.

Notification Options

Admin Groups

admins
win-admins
linux-admins
router-admins

Timeperiods

24x7
workhours
none
us-holidays

Service Groups

ping
http

Host Groups

linux
windows
routers

Service Levels

c - critical
w - warning
u - unknown
r - recovered
f - flapping
s - maintenance

notification_interval

=0
=60
=120

Host Levels

d - down
u - unreachable
r - recovered
f - flapping
s - maintenance

Multi_Level Notifications

Multi_level notifications allow an administrator to perform different levels of notification to different administrators on the same host or service. For example, if there was a service that the administrator wanted to send WARNING level messages to one admin but send CRITICAL level messages to a different administrator they could use this multi_level notification process. The first step in understanding the configuration options is to understand the generic-

contact template. If a service notification was the focus, the options for notification are by default:

```
service_notification_options      w,u,c,r,f,s

define contact{
    name                           generic-contact
    service_notification_period    24x7
    host_notification_period      24x7
    service_notification_options   w,u,c,r,f,s
    host_notification_options     d,u,r,f,s
    service_notification_commands notify-service-by-email
    host_notification_commands    notify-host-by-email
    register                      0
}
```

In order to use multi_level notification you will need to create two separate contacts with different levels of service notification; w (warning) and c (critical). Here joe is created as a contact with the “w” only in the service_notifications_options. Note that even though the generic-contact is used for joe by placing a specific reference to service_notifications_options in this contact information it overrides the default template settings.

```
define contact {
    use                           generic-contact
    contact_name                  joe
    service_notification_options   w
    service_notification_commands notify-service-by-email
    email                         joe@localhost.localdomain
}
```

Bob is created as a contact with the “c” option.

```
define contact {
    use                           generic-contact
    contact_name                  bob
    service_notification_options   c
    service_notification_commands notify-service-by-email
    email                         bob@localhost.localdomain
}
```

Now in order to use these administrators they must be associated with a contactgroup, here a new contactgroup has been created, “linux_admins”.

```
define contactgroup{
    contactgroup_name            linux_admins
    alias                        admin group
    members                      joe,bob
}
```

Once the contactgroup has been created, the group must be associated with the service that you want to have multi-level notifications associated with. In this example, the check_local_users is used with a WARNING state at 2 and a CRITICAL state at 3.

```
define service{
    use                      local-service
    host_name                localhost
    service_description       Current Users
    check_command             check_local_users!2!3
    contact_groups            linux_admins
}
```

Once that has been completed, restart Nagios and then create a situation where the WARNING level and the CRITICAL levels can be attained. In the example you see the process of the soft states created and then the hard state and at that point joe is sent an email for the WARNING state.

```
Jul  9 04:56:04 localhost nagios: SERVICE ALERT: localhost;Current
Users;WARNING;SOFT;1;USERS WARNING - 3 users currently logged in
Jul  9 04:56:14 localhost nagios: SERVICE ALERT: localhost;Current
Users;WARNING;SOFT;2;USERS WARNING - 3 users currently logged in
Jul  9 04:57:14 localhost nagios: SERVICE ALERT: localhost;Current
Users;WARNING;SOFT;3;USERS WARNING - 3 users currently logged in
Jul  9 04:57:14 localhost nagios: SERVICE ALERT: localhost;Current
Users;WARNING;HARD;4;USERS WARNING - 3 users currently logged in
Jul  9 04:57:14 localhost nagios: SERVICE NOTIFICATION: joe;localhost;Current
Users;WARNING;notify-service-by-email;USERS WARNING - 3 users currently logged in
```

Here is the email that only joe receives.

```
From nagios@localhost.localdomain Sat Jul  9 05:07:11 2011
Date: Sat, 9 Jul 2011 05:07:11 -0600
From: nagios@localhost.localdomain
To: joe@localhost.localdomain
Subject: ** PROBLEM Service Alert: localhost/Current Users is WARNING **
***** Nagios *****
Notification Type: PROBLEM
Service: Current Users
Host: localhost
Address: 127.0.0.1
State: WARNING
Date/Time: Sat Jul 9 05:07:11 MDT 2011
Additional Info:
USERS WARNING - 3 users currently logged in
```

Now after creating the CRITICAL state bob is the only one to receive an email.

```
Jul  9 05:02:47 localhost nagios: SERVICE ALERT: localhost;Current
Users;CRITICAL;HARD;4;USERS CRITICAL - 4 users currently logged in
Jul  9 05:02:47 localhost nagios: SERVICE NOTIFICATION: bob;localhost;Current
Users;CRITICAL;notify-service-by-email;USERS CRITICAL - 4 users currently logged
in
```

```

From nagios@localhost.localdomain Sat Jul  9 05:15:34 2011
Date: Sat, 9 Jul 2011 05:15:34 -0600
From: nagios@localhost.localdomain
To: bob@localhost.localdomain
Subject: ** PROBLEM Service Alert: localhost/Current Users is CRITICAL **
***** Nagios *****
Notification Type: PROBLEM
Service: Current Users
Host: localhost
Address: 127.0.0.1
State: CRITICAL
Date/Time: Sat Jul 9 05:15:34 MDT 2011
Additional Info:
USERS CRITICAL - 4 users currently logged in

```

Escalation

Escalation is a process in which if a solution is not produced for a host or service in a specified response time, the problem is referred to the next level. This of course implies that an organization will have a number of levels for administrators. It provides a way to focus resources on those who are capable of solving situations within a reasonable amount of time. Reasonable, is determined by the significance of the host or service that is down.

HOST ESCALATIONS	Admins Responsible	Members	Initial Contact	Escalation
Level 1	Nagios Admins	nagiosadmin, sue, john	1	4
Level 2	Level 2 Engineers	mark,tom	5	8
Level 3	Level 3 Engineers	mary,ralph	9	12

In this example you can see the initial contact will be the Nagios administrators who will normally handle Nagios problems. Once the 5 message is sent out the problem is escalated to the Level 2 Engineers who will become the default notification. Once the 9th message is sent out it will be escalated to Level 3 Engineers who will become the default.

It is important to recognize in the example that Nagios does not measure response in time but rather in the number of messages that are sent out, which are measured on a time interval. The generic-service template for example will notify administrators every 60 minutes.

```
notification_interval      60
```

Messages are sent to contact groups so those groups and the administrators that are a part of those groups must be created.

Setting up the Contacts and Contact Groups

Create all of the users in the htpasswd.users file so they have access to the web interface. This will be important if one

of the administrators leaves a message about the information they found on the problem.

```
cat htpasswd.users
nagiosadmin:fTx/AJMMvBp22
fred:X8agiOot2dRzk
management:0e/oiKAJS0Erc
john:PwI3eSx5QDCp.
sue:YDqeTdQIqn9tE
jim:cW790LhQsU3v6
mark:MCCR2eMPWBx4Y
tom:WK8dJ8ksJeNtU
mary:y4ogaqxCr43OM
ralph:zp5jafw5H2.wA
```

Edit the cgi.cfg file to provide the correct rights so your admins can all fix the same things.

```
authorized_for_system_information=nagiosadmin,john,sue,mark,tom,mary,ralph
authorized_for_configuration_information=nagiosadmin,john,sue,mark,tom,mary,ralph
authorized_for_system_commands=nagiosadmin,john,sue,mark,tom,mary,ralph
authorized_for_all_services=nagiosadmin,management,john,sue,mark,tom,mary,ralph
authorized_for_all_hosts=nagiosadmin,management,john,sue,mark,tom,mary,ralph
authorized_for_all_service_commands=nagiosadmin,john,sue,mark,tom,mary,ralph
authorized_for_all_host_commands=nagiosadmin,john,sue,mark,tom,mary,ralph
```

Create your contacts and provide them with email addresses.

```
define contact{
    contact_name                nagiosadmin
    use                          generic-contact
    alias                        Nagios Admin
    email                        nagios@localhost
}
define contact{
    contact_name                sue
    use                          generic-contact
    alias                        Linux Admin
    email                        sue@localhost
}
```

Define the needed contact groups with the appropriate admins.

```
define contactgroup{
    contactgroup_name        admins
    alias                    Nagios Administrators
    members                  nagiosadmin,sue,john
}
define contactgroup{
    contactgroup_name        engineer2
```

```

alias          Linux Administrators
members        mark,tom
}
define contactgroup{
    contactgroup_name   engineer3
    alias               Linux Administrators
    members            mary,ralph
}

```

When you set up the notification process you can allow for overlap so that two contact groups could be working on the problem.

Create a new configuration file, or place this information in an existing file in the /usr/local/nagios/etc/objects. Here you can see serviceescalation is defined. You will need the host, service_description and then when notification will start and end for this group. You also need to enter the notification_interval.

```

define serviceescalation {
    host_name      bash
    service_description Procs
    first_notification 5
    last_notification 8
    notification_interval 60
    contact_groups engineer2
}
define serviceescalation {
    host_name      bash
    service_description Procs
    first_notification 9
    last_notification 12
    notification_interval 60
    contact_groups engineer3
}

```

If you create notification intervals which overlap, Nagios will use the interval that is the smallest.

Once that is saved and Nagios is restarted you should be able to go to the web interface and select configuration/service escalations and see your changes.

Service Escalations

Service							
Host	Description	Contacts/Groups	First Notification	Last Notification	Notification Interval	Escalation Period	Escalation Options
bash	Procs	engineer3	9	12	1h 0m 0s	24x7	Warning, Unknown, Critical, Recovery
bash	Procs	engineer2	5	8	1h 0m 0s	24x7	Warning, Unknown, Critical, Recovery

When you use escalation_period it is important to realize that the notification_period is not removed, rather the escalation_period must intersect the notification_period. If you have a time period defined as 24x7 as the

notification_period then anything you place in escalation_period will work. However, if the notification_period is a 12x4, illustrated below, and the escalation_period is workhours, illustrated below, escalations only occur for where the two time periods intersect. So you would get escalations only from 09:00-17:00 on Mon-Thurs.

```
define timeperiod{
    timeperiod_name 12x4
    alias            Tech Staff Hours
    monday          06:00-18:00
    tuesday         06:00-18:00
    wednesday       06:00-18:00
    thursday        06:00-18:00
}
define timeperiod{
    timeperiod_name workhours
    alias            Normal Work Hours
    monday          09:00-17:00
    tuesday         09:00-17:00
    wednesday       09:00-17:00
    thursday        09:00-17:00
    friday          09:00-17:00
}
```

You may control the escalation time frame but you will always want to take into account the notification_period first. Here you can see the escalation_period is for workhours.

```
define serviceescalation {
    host_name           localhost
    service_description Current Users
    first_notification 5
    last_notification   8
    notification_interval 60
    escalation_period   workhours
    contact_groups      engineer2
}
define serviceescalation {
    host_name           localhost
    service_description Current Users
    first_notification 9
    last_notification   12
    notification_interval 60
    escalation_period   workhours
    contact_groups      engineer3
}
```

There may be times when you want the escalation to continue until a resolution of the problem. If that is the case then the last_notification must be "0". see the example. The final "last_notification" is changed to "0" instead of "12".

```
define serviceescalation {
    host_name           localhost
```

```

    service_description      Current Users
    first_notification      5
    last_notification       8
    notification_interval   60
    escalation_period       workhours
    contact_groups          engineer2
}
define serviceescalation {
    host_name               localhost
    service_description      Current Users
    first_notification       9
    last_notification        0
    notification_interval   60
    escalation_period        workhours
    contact_groups           engineer3
}

```

Another way to fine tune escalations is when you look at escalation_options. The options that are available are:

Service Options

- w - warning
- u - unknown
- c - critical
- r - recovered
- f - flapping
- s - scheduled maintenance

Host Options

- d - down
- u - unreachable
- r - recovered
- f - flapping
- s - scheduled maintenance

The example below shows that the warning, flapping and scheduled maintenance have been removed for this service.

```

define serviceescalation {
    host_name               localhost
    service_description      Current Users
    first_notification       5
    last_notification        8
    notification_interval   60
    escalation_period        workhours
    escalation_options       u,c,r
    contact_groups           engineer2
}
define serviceescalation {
    host_name               localhost

```

```

service_description      Current Users
first_notification      9
last_notification        0
notification_interval   60
escalation_period       workhours
escalation_options      u,c,r
contact_groups          engineer3
}

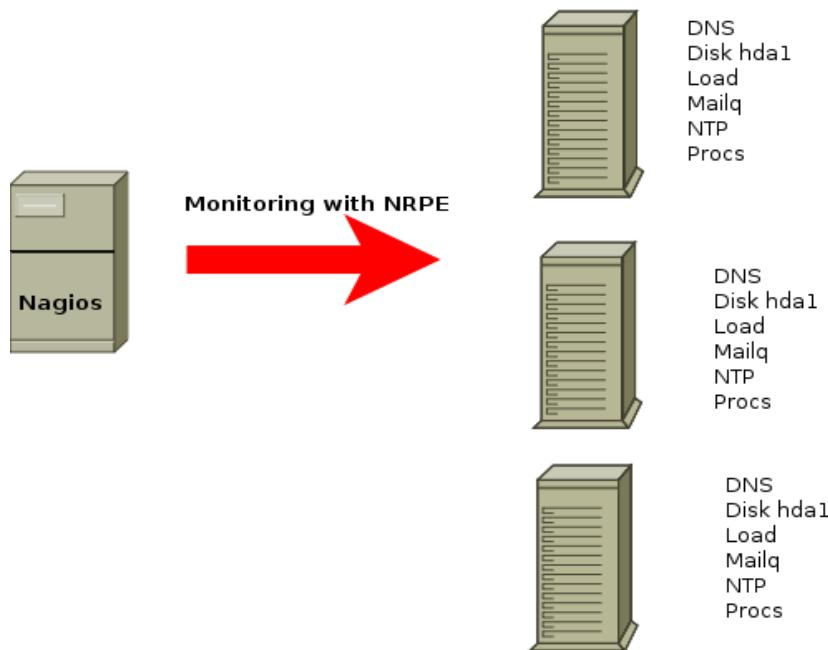
```

Notification: Host and Service Dependencies

A major factor of frustration for administrators is when checks depend upon a service or host and that when the host or service goes down administrators receive notifications for all of the dependent hosts and services. This does not help administrators solve the real problem and in fact directs the administrator in the wrong direction. For example, if you were monitoring your Linux network using NRPE and the NRPE service failed, in addition to getting notified about the NRPE service failure, you would receive notification from all of the services network wide you were monitoring....ugly.

This problem can be eliminated by implementing host and service dependencies. In the example, all of the Linux servers being monitored by NRPE are dependent upon NRPE. So the idea is to monitor the NRPE process to verify that it is working and to add all of the NRPE checks as dependents.

NRPE Failure Generates 19 Notifications



The first thing you need to do is create a command definition for checking the NRPE process. Here a simple command will determine if the process is functioning correctly.

```
define command {
    command_name      nrpe_verify
    command_line      $USER1$/check_nrpe -H $HOSTADDRESS$}
}
```

Next you need to create a service definition.

```
define service{
    host_name          bash
    service_description NRPE
    check_command      nrpe_verify
}
}
```

You will need to set up the service or host dependencies which are related. Here you can see that the one server has listed 5 services which are dependent upon NRPE.

```
##### Dependency Checks #####
define servicedependency{
    host_name          bash
    service_description NRPE
    dependent_host_name bash
    dependent_service_description DNS,Load,Mailq,NTP,Procs
    notification_failure_criteria  c,u
    execution_failure_criteria     n
}
}
```

By setting up these dependencies it will eliminate the unnecessary notifications.

Chapter 5: Management

Management of the Nagios server can often be performed from the web interface. Users are able to view, acknowledge problems and communicate situations using the web interface.

Web Interface

The web interface of Nagios Core provides a menu on the left with specific details in the main body of the page which contains more clickable links to information.

The screenshot shows the Nagios Core web interface. On the left, there is a navigation sidebar with links for General, Current Status, Reports, and System. The main content area displays the 'Current Network Status' and 'Service Status Details For Host 'bash''. The 'Service Status Details' table lists various services with their status, last check time, duration, attempt count, and status information. The table includes rows for Check Postfix, DNS, Disk sda1, Load, Mailq, NRPE, NTP, Procs, SSH Check Disk, SSH Check Load, SSH Check Users, SSH Check_Multi, SSH Check_Net_Connections, SSH Check_Processes, SSH Check_Zombies, SSH Current Web Servers, SSH DNS for Nagios, Users, Wordpress_Database, and Zombies.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
bash	Check Postfix	OK	11-29-2011 04:32:25	20d 13h 49m 37s	1/3	No Postfix Fatal Errors
	DNS	OK	11-29-2011 04:35:26	21d 17h 2m 41s	1/3	DNS OK: 0.059 seconds response time. google.com returns 173.194.33.16,173.194.33.17,173.194.33.18,173.194.33.19,173.194.33.20
	Disk sda1	OK	11-29-2011 04:35:35	21d 17h 1m 36s	1/3	DISK OK - free space: / 1823 MB (60% inode=78%):
	Load	OK	11-29-2011 04:37:00	21d 17h 0m 31s	1/3	OK - load average: 0.00, 0.00, 0.00
	Mailq	OK	11-29-2011 04:39:35	21d 16h 59m 25s	1/3	OK: mailq reports queue is empty
	NRPE	OK	11-29-2011 04:40:46	21d 16h 58m 20s	1/3	NRPE v2.12
	NTP	OK	11-29-2011 04:32:48	13d 17h 19m 2s	1/3	NTP OK: Offset 0.8933043697 secs
	Procs	OK	11-29-2011 04:32:31	21d 16h 56m 10s	1/3	PROCS OK: 19 processes
	SSH Check Disk	OK	11-29-2011 04:33:54	18d 23h 37m 3s	1/3	DISK OK - free space: / 1823 MB (60% inode=78%):
	SSH Check Load	OK	11-29-2011 04:35:36	18d 23h 36m 8s	1/3	OK - load average: 0.00, 0.00, 0.00
	SSH Check Users	OK	11-29-2011 04:37:00	18d 23h 35m 13s	1/3	USERS OK - 0 users currently logged in
	SSH Check_Multi	OK	11-29-2011 04:32:59	0d 8h 38m 50s	1/3	OK - 16 plugins checked, 16 ok
	SSH Check_Net_Connections	OK	11-29-2011 04:40:53	17d 12h 47m 46s	1/3	192.168.5.163
	SSH Check_Processes	OK	11-29-2011 04:40:52	18d 23h 32m 29s	1/3	PROCS OK: 20 processes
	SSH Check_Zombies	OK	11-29-2011 04:32:37	18d 23h 31m 34s	1/3	PROCS OK: 0 processes with STATE = Z
	SSH Current Web Servers	OK	11-29-2011 04:34:00	18d 20h 17m 20s	1/3	OK - 1 plugins checked, 1 ok
	SSH DNS for Nagios	OK	11-29-2011 04:35:36	18d 1h 28m 39s	1/3	DNS OK: 0.074 seconds response time. nagios.org returns 173.45.235.65
	Users	OK	11-29-2011 04:37:00	18d 14h 7m 5s	1/3	USERS OK - 0 users currently logged in
	Wordpress_Database	CRITICAL	11-29-2011 04:32:23	21d 16h 53m 59s	3/3	Host '192.168.5.163' is not allowed to connect to this MySQL server
	Zombies	OK	11-29-2011 04:39:34	21d 17h 3m 33s	1/3	PROCS OK: 0 processes with STATE = Z

Home

The “Home” link provides access to the main page which provides the current version with a link to check on updates as well as links to training and certification options, news items, tutorials and links to Nagios plugins at Nagios Exchange. This page will keep you up to date on the changes that happen with Nagios.

Nagios® Core™

Nagios® Core™ Version 3.3.1
July 25, 2011
[Check for updates](#)

Get Started

- Start monitoring your infrastructure
- Change the look and feel of Nagios
- Extend Nagios with hundreds of addons
- Get support
- Get training
- Get certified

Don't Miss...

-  Attend the Nagios World Conference September 27th-29th, 2011.
Saint Paul, MN | September 27-29, 2011
- Try the new Nagios V-Shell interface
- Monitor business processes with the new Nagios BPI addon.

Quick Links

- Nagios Library (tutorials and docs)
- Nagios Labs (development blog)
- Nagios Exchange (plugins and addons)

Latest News

- Nagios Conference Sold Out!
- Conference Packages Nearly Sold Out!
- Nagios Is Hiring! Join Our Team As A Software Engineer
- More news...

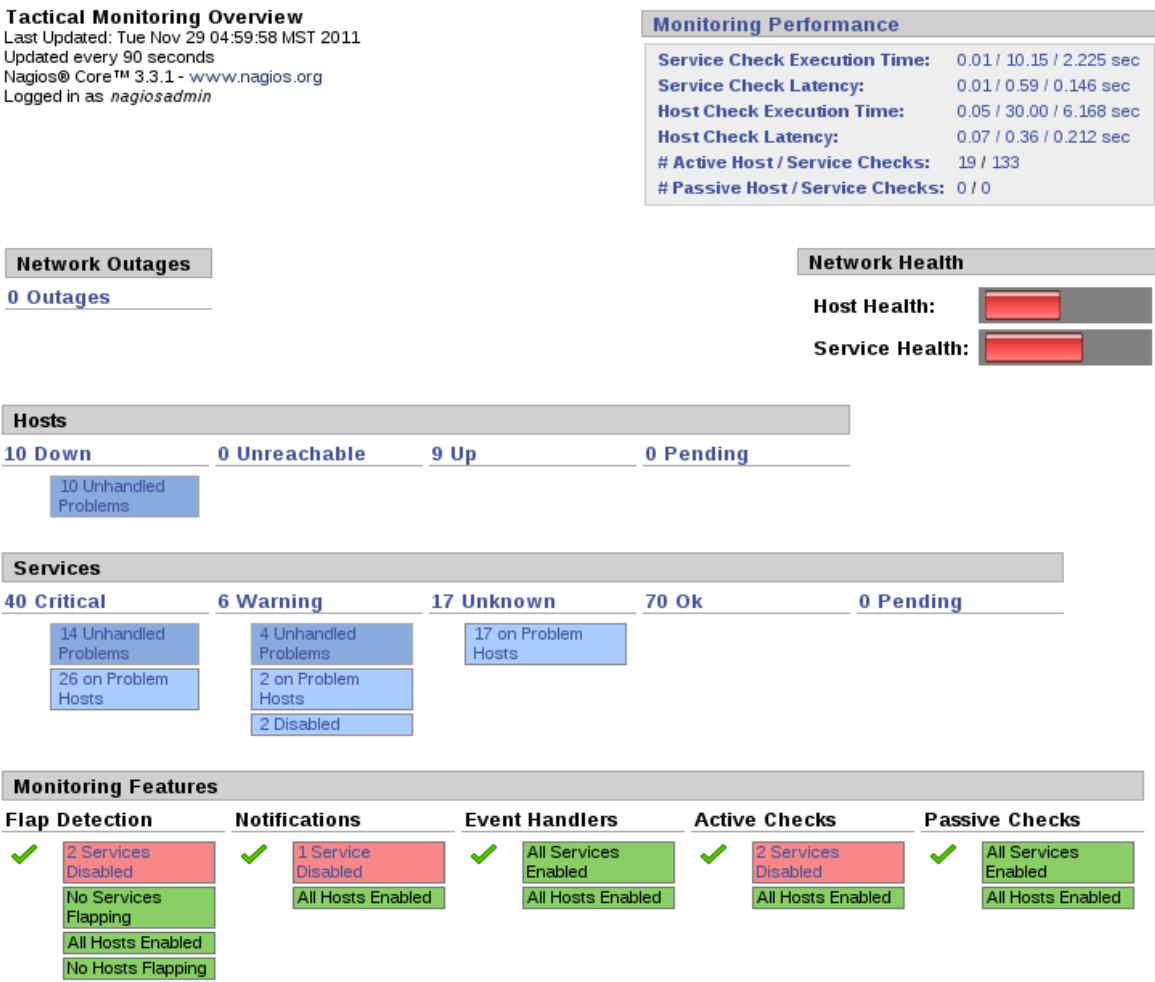
Documentation

This link will take you to the Nagios site where you will find manuals, tutorials and links to other information about Nagios.

The screenshot shows the Nagios Library page. At the top, there's a navigation bar with links for Home, News, Library, Training, and Subscribe. The Library link is highlighted. To the right of the navigation is a "Library" icon (stack of books) and the word "Library". Below the navigation, there are links for Home, Library, Product Sections, Nagios Core, and Manuals. The main content area has a heading "Nagios Core Manuals" with "Available Manuals" below it. It says "Online manuals are available for the most recent versions of Nagios Core and key Nagios addons." There's an image of a stack of books. Below this, under "Nagios Core 3.x", it says "Nagios Core documentation is updated daily." with links for "Online (HTML) Manual" and "PDF Manual". Under "NRPE", there's a link for "PDF Manual". Under "NDOUtils", it says "Installation Manual:" with a "PDF Manual" link, and "Database Design Guide:" with a "PDF Manual" link. On the right side, there's a sidebar titled "Product Sections" with links for Nagios XI, Nagios Fusion, Nagios Core, Documentation, and Manuals (which is currently selected). Other sections include Video Tutorials, Special Topics, and Tech Tips. There's also a search bar labeled "Search the Library" and a "Library Login" section with fields for Username, Password, and Remember Me, along with a "Login" button.

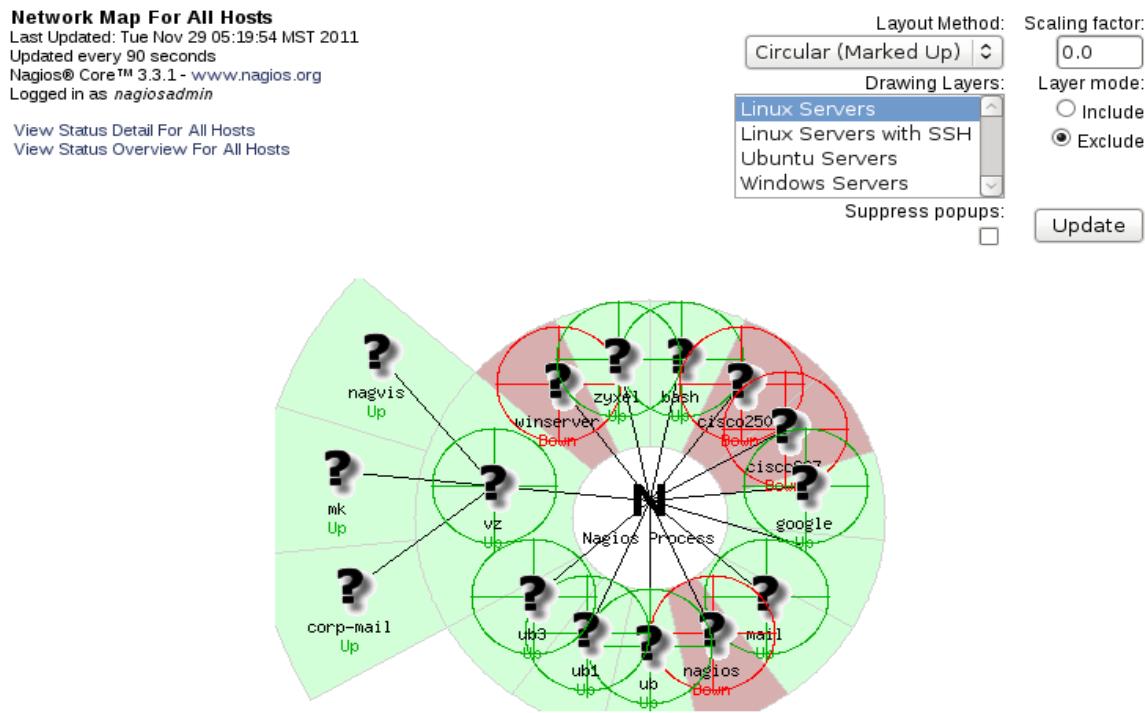
Tactical Overview

The “Tactical Overview” is designed to provide an overall look at the health of not only the Nagios server in the “Monitoring Performance” but also the entire network that is being monitored including all of the hosts and services. This information is updated every 90 seconds to provide a fresh look. Each window represents a different category on your network; Monitoring Performance (Nagios server resources), Network Outages, Network Health (summary), Hosts, Services and Monitoring Features (features currently available). Most of these are links to more detailed information.

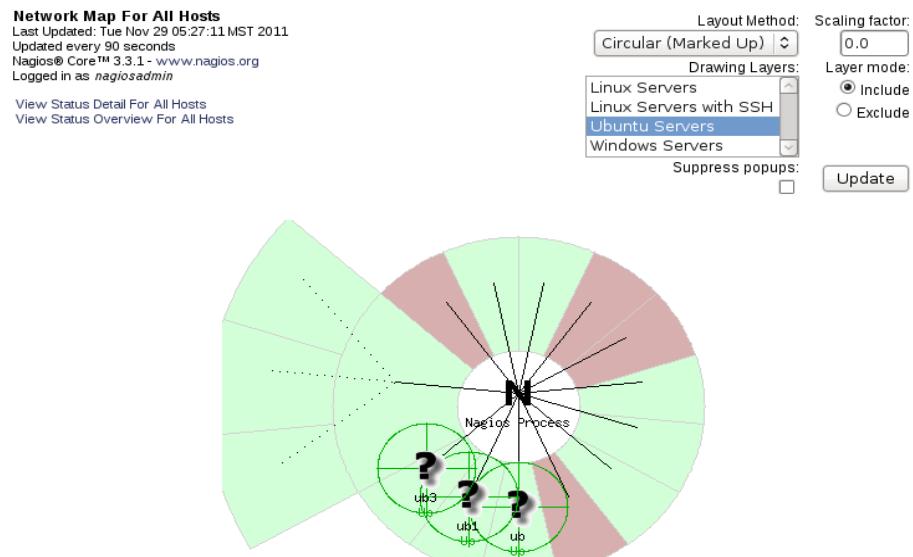


Map

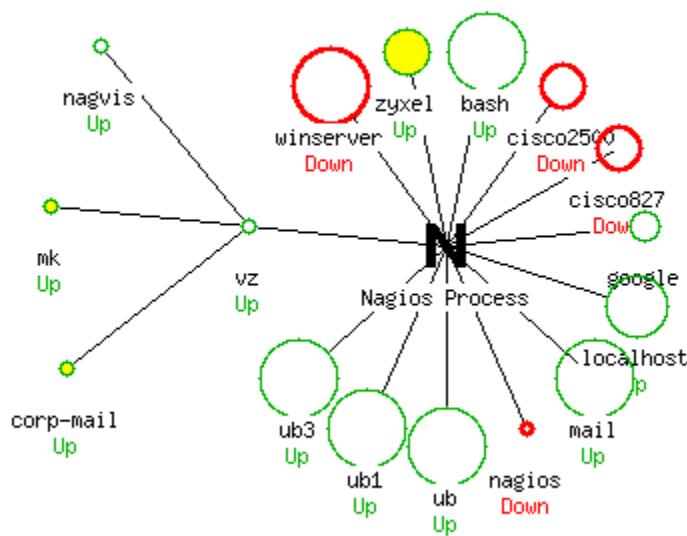
The “Map” represents ping connections to each of the hosts being monitored on the network. The map provides a Layout Method which basically provides different views of the hosts. The “Drawing Layers” allows the selection of different host groups. The “Scaling factor” allows for size representation on the image.



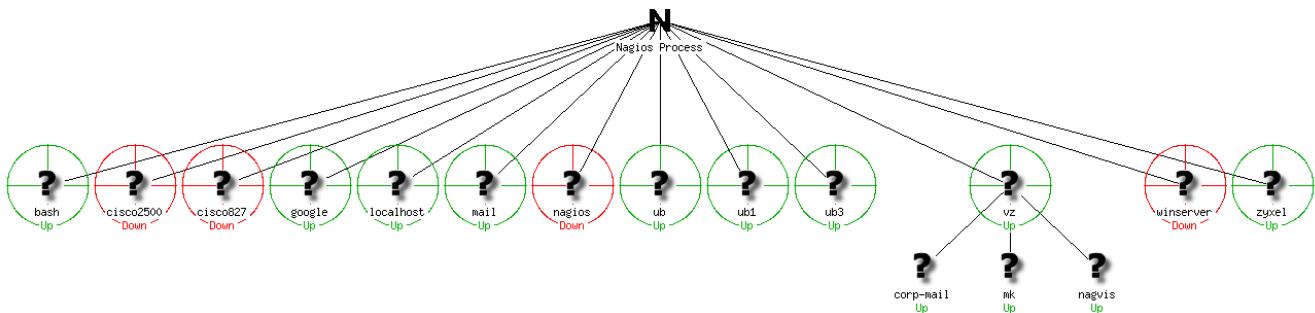
The map has options to change how it looks and to select or deselect (Include or Exclude) categories of hosts. In this example Ubuntu Servers is selected and the “Include” is used to only show Ubuntu servers.



Here is an example of the Circular (Ballon).



The Balanced Tree shows the same hosts and levels. The levels of hosts represent the parent relationships between hosts. For example, if you had a virtual server hosting containers as you can see in this example. Or it may represent the relationships of hosts, routers and switches.



Hosts

The “Hosts” link provides access to each of the hosts which are being monitored on the network. At the top of the page is a summary of the network conditions. There are links to more specific details for host groups, and how to view them in different ways. There are also two summary windows for hosts status and service status on the hosts.

Current Network Status
 Last Updated: Tue Nov 29 06:06:34 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1- www.nagios.org
 Logged in as *nagiosadmin*

[View Service Status Detail For All Host Groups](#)
[View Status Overview For All Host Groups](#)
[View Status Summary For All Host Groups](#)
[View Status Grid For All Host Groups](#)



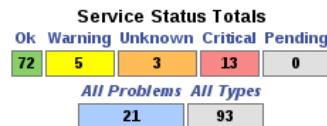
Host Status Details For All Host Groups

Host ↑↓	Status ↑↓	Last Check ↑↓	Duration ↑↓	Status Information
bash		UP	11-29-2011 06:05:10	21d 18h 28m 31s
corp-mail		UP	11-29-2011 06:05:50	0d 0h 52m 11s
google		UP	11-29-2011 06:06:22	3d 23h 2m 57s
localhost		UP	11-29-2011 06:01:30	19d 18h 50m 21s
mail		UP	11-29-2011 06:01:20	48d 20h 50m 8s
mk		UP	11-29-2011 06:01:30	0d 0h 51m 41s

The link “Service Status Detail” for the hosts provide the complete list of all hosts and all services on those hosts.

Current Network Status
 Last Updated: Tue Nov 29 06:09:03 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1- www.nagios.org
 Logged in as *nagiosadmin*

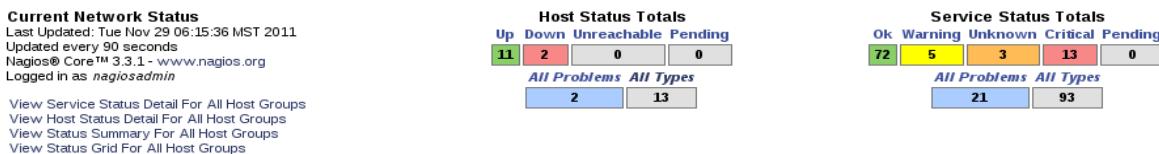
[View Host Status Detail For All Host Groups](#)
[View Status Overview For All Host Groups](#)
[View Status Summary For All Host Groups](#)
[View Status Grid For All Host Groups](#)



Service Status Details For All Host Groups

Host ↑↓	Service ↑↓	Status ↑↓	Last Check ↑↓	Duration ↑↓	Attempt ↑↓	Status Information
bash	Check Postfix		OK	11-29-2011 06:02:25	20d 15h 16m 51s	1/3
	DNS		OK	11-29-2011 06:05:26	21d 18h 29m 55s	1/3
	Disk sda1		OK	11-29-2011 06:05:35	21d 18h 28m 50s	1/3
	Load		OK	11-29-2011 06:07:00	21d 18h 27m 45s	1/3
	Mailq		OK	11-29-2011 05:59:35	21d 18h 26m 39s	1/3
	NRPE		OK	11-29-2011 06:00:46	21d 18h 25m 34s	1/3
	NTP		OK	11-29-2011 06:02:48	13d 18h 46m 16s	1/3
	Procs		OK	11-29-2011 06:02:31	21d 18h 23m 24s	1/3
	SNMP Apache Processes		WARNING	11-29-2011 06:03:53	0d 0h 49m 10s	3/3
	SNMP CPU usage		OK	11-29-2011 06:01:06	0d 0h 47m 57s	1/3

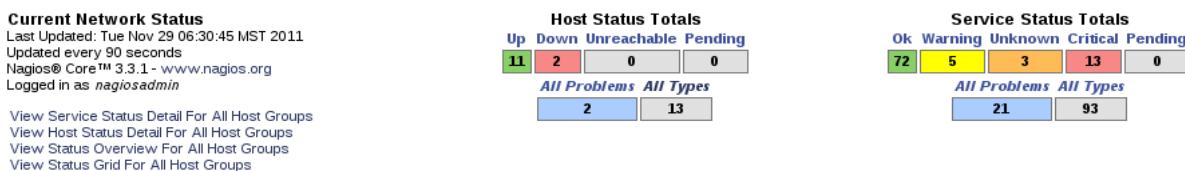
“Status Overview” provides a graphical summary of host groups and services.



Service Overview For All Host Groups



“Status Summary” is a more compact way of viewing similar information. More information can be gained by clicking on the links in the information.



Status Summary For All Host Groups

Host Group	Host Status Summary	Service Status Summary
Linux Servers (linux-servers)	1 UP	7 OK
Linux Servers with SSH (ssh-checks)	1 UP	23 OK 1 WARNING : 1 Unhandled
Ubuntu Servers (ubuntu_servers)	3 UP	17 OK 13 CRITICAL : 13 Unhandled
Windows Servers (windows-servers)	1 DOWN : 1 Unhandled	3 UNKNOWN : 3 on Problem Hosts

Finally, there is the option to view the status in a grid.

Status Grid For All Host Groups

Host	Services	Linux Servers (linux-servers)										Actions
		localhost	Current Load	Current Users	HTTP	PING	Root Partition	SSH	Total Processes			
Linux Servers with SSH (ssh-checks)												
bash	Check Postfix DNS Disk sda1 Load Mailq NRPE NTP Procs SNMP Apache Processes SNMP CPU usage SNMP Check Disk Space / SNMP ETH0 Traffic SNMP Memory usage SSH Check Disk SSH Check Load SSH Check_Multi SSH Check_Net_Connections SSH Check_Proceses SSH Check_Zombies SSH Current Web Servers SSH DNS for Nagios Users Zombies											

Services

The services can be viewed altogether by selecting the “Services” link in the menu. The “Host Status” and “Service Status” summaries are at the top with additional links for a history of the hosts, notifications sent and host detail information.



For each service listed you see the host that the service check was on and the name of the “Service” which is a link for more detailed information. This includes links to service details and service commands that can be executed.

The screenshot shows the detailed view for a specific service. On the left, under "Service Information", is a summary of the service's status and configuration. On the right, under "Service Load", are details about the host it runs on. Below these are sections for "Service State Information" and "Service Commands".

Service Information:

- Last Updated: Tue Nov 29 06:43:58 MST 2011
- Updated every 90 seconds
- Nagios® Core™ 3.3.1 - www.nagios.org
- Logged in as nagiosadmin

Service Load:

- On Host: Company Server (bash)
- Member of: No servicegroups.
- IP Address: 192.168.5.190

Service State Information:

Current Status:	OK (for 21d 19h 2m 40s)
Status Information:	OK - load average: 0.02, 0.02, 0.00
Performance Data:	load1=0.020;15.000;30.000;0; load5=0.020;10.000;25.000;0; load15=0.000;5.000;20.000;0;
Current Attempt:	1/3 (HARD state)
Last Check Time:	11-29-2011 06:37:00
Check Type:	ACTIVE
Check Latency / Duration:	0.114 / 0.043 seconds
Next Scheduled Check:	11-29-2011 06:47:00
Last State Change:	11-07-2011 11:41:18
Last Notification:	N/A (notification 0)
Is This Service Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	11-29-2011 06:43:48 (0d 0h 0m 10s ago)

Service Commands:

- Disable active checks of this service
- Re-schedule the next check of this service
- Submit passive check result for this service
- Stop accepting passive checks for this service
- Stop obsessing over this service
- Disable notifications for this service
- Send custom service notification
- Schedule downtime for this service
- Disable event handler for this service
- Disable flap detection for this service

Service Comments:

Add a new comment | Delete all comments

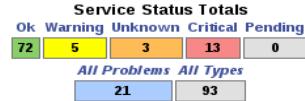
Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
This service has no comments associated with it							

Host Groups

The “Host Groups” provides 3 views, status, summary and grid.

Current Network Status
 Last Updated: Tue Nov 29 06:15:36 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

[View Service Status Detail For All Host Groups](#)
[View Host Status Detail For All Host Groups](#)
[View Status Summary For All Host Groups](#)
[View Status Grid For All Host Groups](#)



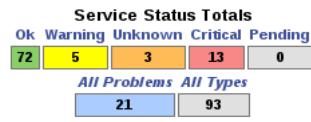
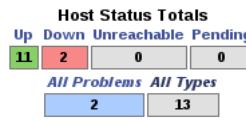
Service Overview For All Host Groups

Linux Servers (linux-servers)				Linux Servers with SSH (ssh-checks)				Ubuntu Servers (ubuntu_servers)			
Host	Status	Services	Actions	Host	Status	Services	Actions	Host	Status	Services	Actions
localhost	UP	7 OK		bash	UP	23 OK		ub	UP	9 OK	
						1 WARNING				1 CRITICAL	

“Status Summary” is a more compact way of viewing similar information. More information can be gained by clicking on the links in the information.

Current Network Status
 Last Updated: Tue Nov 29 06:30:45 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

[View Service Status Detail For All Host Groups](#)
[View Host Status Detail For All Host Groups](#)
[View Status Overview For All Host Groups](#)
[View Status Grid For All Host Groups](#)



Status Summary For All Host Groups

Host Group	Host Status Summary	Service Status Summary
Linux Servers (linux-servers)	1 UP	7 OK
Linux Servers with SSH (ssh-checks)	1 UP	23 OK 1 WARNING : 1 Unhandled
Ubuntu Servers (ubuntu_servers)	3 UP	17 OK 13 CRITICAL : 13 Unhandled
Windows Servers (windows-servers)	1 DOWN : 1 Unhandled	3 UNKNOWN : 3 on Problem Hosts

Finally, there is the option to view the status in a grid.

Status Grid For All Host Groups

Linux Servers (linux-servers)																					
Host	Services	Actions																			
localhost	Current Load Current Users HTTP PING Root Partition SSH Total Processes																				
Linux Servers with SSH (ssh-checks)																					
bash	Check Postfix DNS Disk sda1 Load Mailq NPPe NTP Procs SNMP Apache Processes SNMP CPU usage SNMP Check Disk Space / SNMP ETH0 Traffic SNMP Memory usage SSH Check Disk SSH Check Load SSH Check Users SSH Check Multi SSH Check Net_Connections SSH Check Processes SSH Check Zombies SSH Current Web Servers SSH DNS for Nagios Users Zombies																				

Service Groups

The “Service Groups” provides 3 views, status, summary and grid.

Current Network Status
 Last Updated: Tue Nov 29 06:49:45 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

[View Service Status Detail For All Service Groups](#)
[View Status Summary For All Service Groups](#)
[View Service Status Grid For All Service Groups](#)



Service Overview For All Service Groups

Web Servers (web)

Host	Status	Services	Actions
ub	UP	1 OK	
ub1	UP	1 OK	
ub3	UP	1 OK	

Status Summary For All Service Groups

Service Group	Host Status Summary	Service Status Summary
Web Servers (web)	3 UP	3 OK

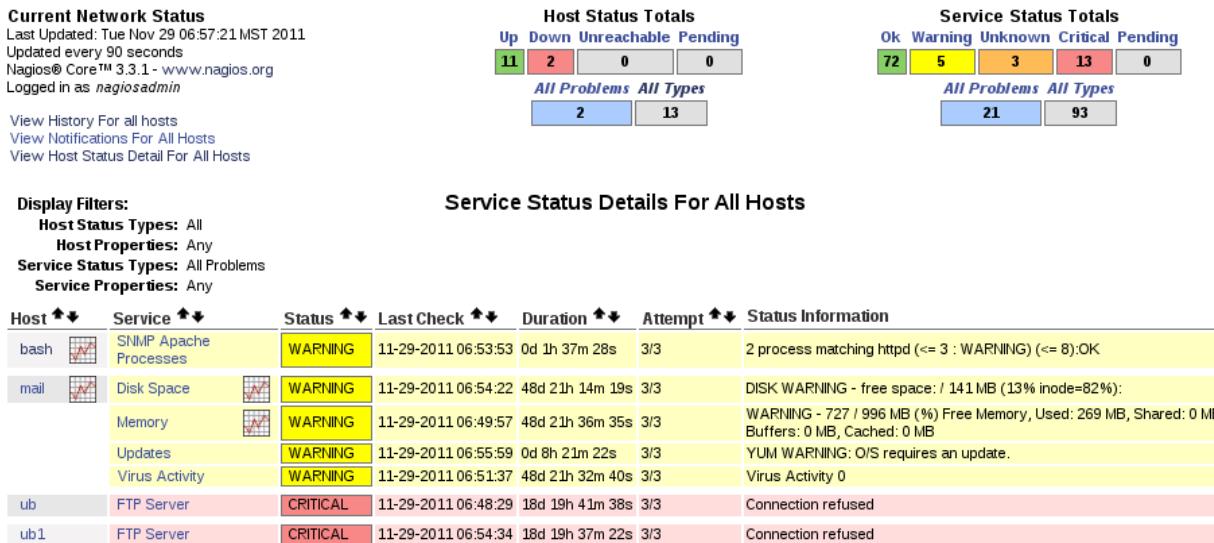
Status Grid For All Service Groups

Web Servers (web)

Host	Services	Actions
ub	HTTP	
ub1	HTTP	
ub3	HTTP	

Problems

Detecting problems and making notification to administrators is the heart of what Nagios can do for you. The basic view provides only problems related to hosts, whether they are WARNING states or CRITICAL states. If you click on the specific problem you will be taken to greater detail about the problem.



If you select the history link you will see what has happened historically with problems. Note that you can make modifications concerning what you see and how you see it with the options on the right.



November 29, 2011 06:00

```
[11-29-2011 06:05:07] Nagios 3.3.1 starting... (PID=21937)
[11-29-2011 06:05:07] Caught SIGHUP, restarting...
[11-29-2011 06:03:46] Nagios 3.3.1 starting... (PID=21937)
[11-29-2011 06:03:46] Caught SIGHUP, restarting...
[11-29-2011 06:03:22] Nagios 3.3.1 starting... (PID=21937)
[11-29-2011 06:03:22] Caught SIGHUP, restarting...
[11-29-2011 06:01:32] SERVICE ALERT: bash,SNMP ETH0 Traffic;OK;HARD;3;venet0:UP:OK
[11-29-2011 06:01:10] Nagios 3.3.1 starting... (PID=21937)
[11-29-2011 06:01:10] Caught SIGHUP, restarting...
```

The notifications link takes you to a page which shows each notification and the reason the notification was sent. Note that it provides information about how the notification was made as well. This will be especially helpful in tracking escalation using various methods of communication.

Host	Service	Type	Time	Contact	Notification Command	Information
ub3	IMAP Server	CRITICAL	11-29-2011 07:00:38	nagiosadmin	notify-service-by-email	Connection refused
ub3	IMAP Server	CRITICAL	11-29-2011 07:00:38	sue	notify-service-by-email	Connection refused
ub3	IMAP Server	CRITICAL	11-29-2011 07:00:38	john	notify-service-by-email	Connection refused

The status detail link on the original “Problems” page simply provides access information about each host.

The “Problems” section in the menu also provides links to all hosts or services or network outages.

Quick Search

Once you start getting the web interface loaded with many hosts and services the “Quick Search” link can be very helpful in finding hosts that are monitored on the network. Here the search for a host named “mail” shows results.

Host	Service	Status	Last Check	Duration	Attempt	Status Information
mail	Amavis: Virus Protection	OK	11-29-2011 07:12:10	48d 21h 57m 41s	1/3	Amavis is Running
	CPU	OK	11-29-2011 07:07:04	48d 21h 57m 15s	1/3	CPU STATISTICS OK: user=0.00% system=0.30% iowait=0.00% idle=99.70%
	Check Imap Content	OK	11-29-2011 07:04:18	22d 0h 14m 50s	1/3	IMAP RECEIVE OK - 0 seconds, 5 found, 0 captured

Availability

Availability provides a way to see the average uptime of a system over a period of time. This is a common request for evaluating performance and creating a report. “Availability” which is part of the “Reports” section has several steps to take in making a selection of the information required. The first option allows you to select from; Hostgroup, Host, Servicegroup or Service.

Availability Report
Last Updated: Tue Nov 29 07:17:00 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 1: Select Report Type

Type:

The second step allows you to select all of the hosts or services or to limit the output to a specific host or service.

Availability Report
Last Updated: Tue Nov 29 07:22:06 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 2: Select Host

Host(s):

*Tip: If you want to have the option of getting the availability data in CSV format, select ** ALL HOSTS ** from the pull-down menu.*

Step three allows you to select the detail that you need to review for the report. These specifics include choosing time parameters for the information, state information and assumptions and the ability to scan previous reports in order to get a better picture of a problem developing over time.

Host Availability Report
Last Updated: Tue Nov 29 07:24:07 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 3: Select Report Options

Report Period:

If Custom Report Period...

Start Date (Inclusive):

End Date (Inclusive):

Report time Period:

Assume Initial States:

Assume State Retention:

Assume States During Program Downtime:

Include Soft States:

First Assumed Host State:

First Assumed Service State:

Backtracked Archives (To Scan For Initial States):

The output of the report breaks down the information into host and service details. The host details provides “State”, the reason the state was reported, the time of the state as well as the percentage of time the host was in that state. The service detail focuses on the time and percentage a service was in a particular state.

Host Availability Report
Last Updated: Tue Nov 29 07:26:07 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as **nagiosadmin**

[View Availability Report For All Hosts](#)
[View Trends For This Host](#)
[View Alert Histogram For This Host](#)
[View Status Detail For This Host](#)
[View Alert History For This Host](#)
[View Notifications For This Host](#)

Host 'bash'

11-29-2011 00:00:00 to 11-29-2011 07:26:07
Duration: 0d 7h 26m 7s

First assumed host state: Unspecified First assumed service state: Unspecified
Report period: Today Backtracked archives: 4
Update

[Availability report completed in 0 min 0 sec]

Host State Breakdowns:

State	Type / Reason	Time	% Total Time	% Known Time
UP	Unscheduled	0d 7h 26m 7s	100.000%	100.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	0d 7h 26m 7s	100.000%	100.000%
DOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	0d 0h 0m 0s	0.000%	0.000%
UNREACHABLE	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	0d 0h 0m 0s	0.000%	0.000%
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	0d 7h 26m 7s	100.000%	100.000%

State Breakdowns For Host Services:

Service	% Time OK	% Time Warning	% Time Unknown	% Time Critical	% Time Undetermined
Check Postfix	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
DNS	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
Disk sda1	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
Load	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
Mailq	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
NRPE	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
NTP	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
Procs	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
SNMP Apache Processes	0.000% (0.000%)	27.373% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	72.627%
child processes	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	100.000%

Trends

Reviewing “Trends” is a way to review information that may be helpful in determining problems which are developing over time or limitations which are developing for the network or hosts. The first step provides a choice of service or host.

Host and Service State Trends
Last Updated: Tue Nov 29 07:44:02 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as **nagiosadmin**

Step 1: Select Report Type

Type: Host

[Continue to Step 2](#)

The second step allows you to choose which host or service.

Host and Service State Trends
Last Updated: Tue Nov 29 07:47:29 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 2: Select Host

Host:

In the final step you can choose date limits, state assumptions and scan back archives for a better understanding of the trend.

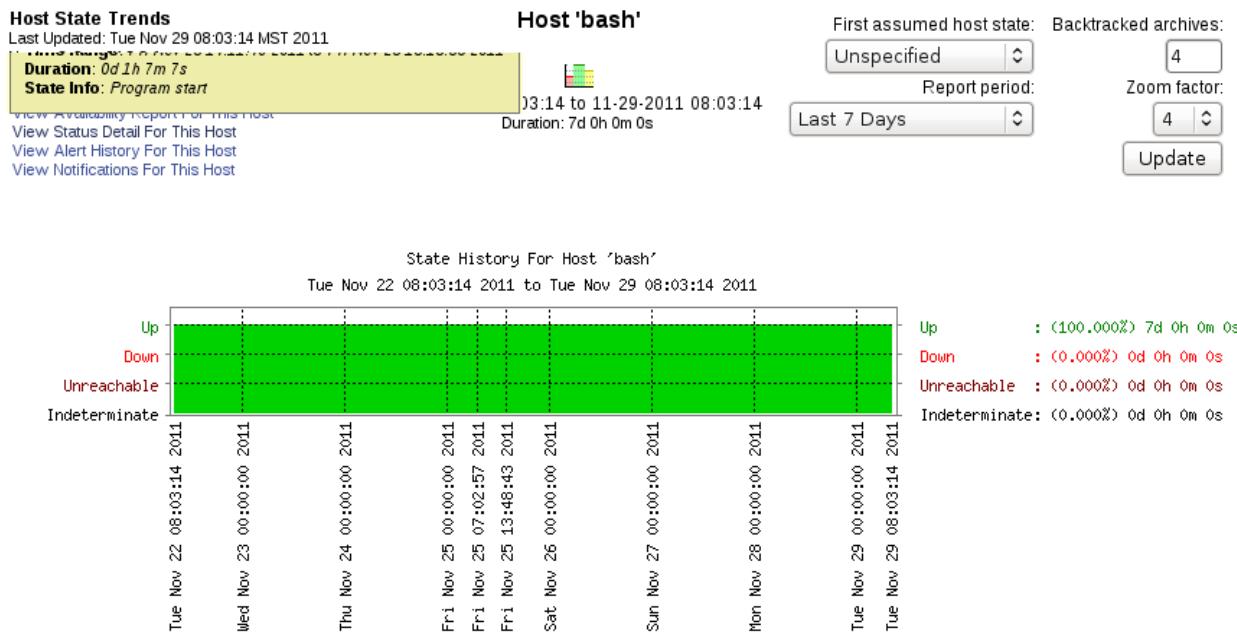
Host State Trends
Last Updated: Tue Nov 29 07:50:44 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 3: Select Report Options

Report period:
If Custom Report Period...
Start Date (Inclusive):
End Date (Inclusive):

Assume Initial States:
Assume State Retention:
Assume States During Program Downtime:
Include Soft States:
First Assumed Host State:
Backtracked Archives (To Scan For Initial States):
Suppress image map:
Suppress popups:

The trends report then provides the data in a chart format to help understand the trends.



Alerts

The “Alerts” section provides a history of alerts that have been sent out. You can choose based on the type of alert(soft or hard state), or the specific alerts you want to see like host, service, warning, critical, etc. Those options work for the history.



November 29, 2011 06:00

- [11-29-2011 06:05:07] Nagios 3.3.1 starting... (PID=21937)
- [11-29-2011 06:05:07] Caught SIGHUP, restarting...
- [11-29-2011 06:03:46] Nagios 3.3.1 starting... (PID=21937)
- [11-29-2011 06:03:46] Caught SIGHUP, restarting...
- [11-29-2011 06:03:22] Nagios 3.3.1 starting... (PID=21937)
- [11-29-2011 06:03:22] Caught SIGHUP, restarting...
- [11-29-2011 06:01:32] SERVICE ALERT: bash,SNMP ETH0 Traffic:OK;HARD;3;venet0:UP:1 UP: OK

The “Summary” for alerts allows you to configure the output of the alert information and limit it to a timeperiod, or

specific hosts, services that you need to review. This shows most recent alerts, top alert producers and alert totals.

Alert Summary Report
 Last Updated: Tue Nov 29 13:08:34 MST 2011
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Standard Reports:
 Report Type: **25 Most Recent Hard Alerts**

Custom Report Options:
 Report Type: **Most Recent Alerts**
 Report Period: **Last 7 Days**
 If Custom Report Period...
 Start Date (Inclusive): **November 1 2011**
 End Date (Inclusive): **November 29 2011**
 Limit To Hostgroup: **** ALL HOSTGROUPS ****
 Limit To Servicegroup: **** ALL SERVICEGROUPS ****
 Limit To Host: **** ALL HOSTS ****
 Alert Types: **Host and Service Alerts**
 State Types: **Hard and Soft States**
 Host States: **All Host States**
 Service States: **All Service States**
 Max List Items: **25**

Here is the output of the summary.

Alert Summary Report
 Last Updated: Tue Nov 29 13:11:06 MST 2011
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Most Recent Alerts							
11-22-2011 13:11:06 to 11-29-2011 13:11:06 Duration: 7d 0h 0m 0s							
Displaying most recent 25 of 124 total matching alerts							
Time	Alert Type	Host	Service	State	State Type	Information	
11-29-2011 11:26:08	Service Alert	mail	Updates	WARNING	HARD	YUM WARNING: O/S requires an update.	
11-29-2011 11:16:18	Service Alert	mail	Updates	CRITICAL	HARD	CHECK_NRPE: Socket timeout after 10 seconds.	

Report Options Summary:
Alert Types: Host & Service Alerts
State Types: Soft & Hard States
Host States: Up, Down, Unreachable
Service States: Ok, Warning, Unknown, Critical

The “Histogram” creates a historic graph of the alerts you select. This is a three step process. In the first step select host or service.

Host and Service Alert Histogram
Last Updated: Tue Nov 29 13:12:17 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 1: Select Report Type

Type:

In the second step select the specific host or service you would like to see.

Host and Service Alert Histogram
Last Updated: Tue Nov 29 13:12:48 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 2: Select Host

Host:

The third step allows for a lot of fine tuning including time period, hosts and states.

Host Alert Histogram
Last Updated: Tue Nov 29 13:13:11 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 3: Select Report Options

Report Period:

If Custom Report Period...

Start Date (Inclusive):

End Date (Inclusive):

Statistics Breakdown:

Events To Graph:

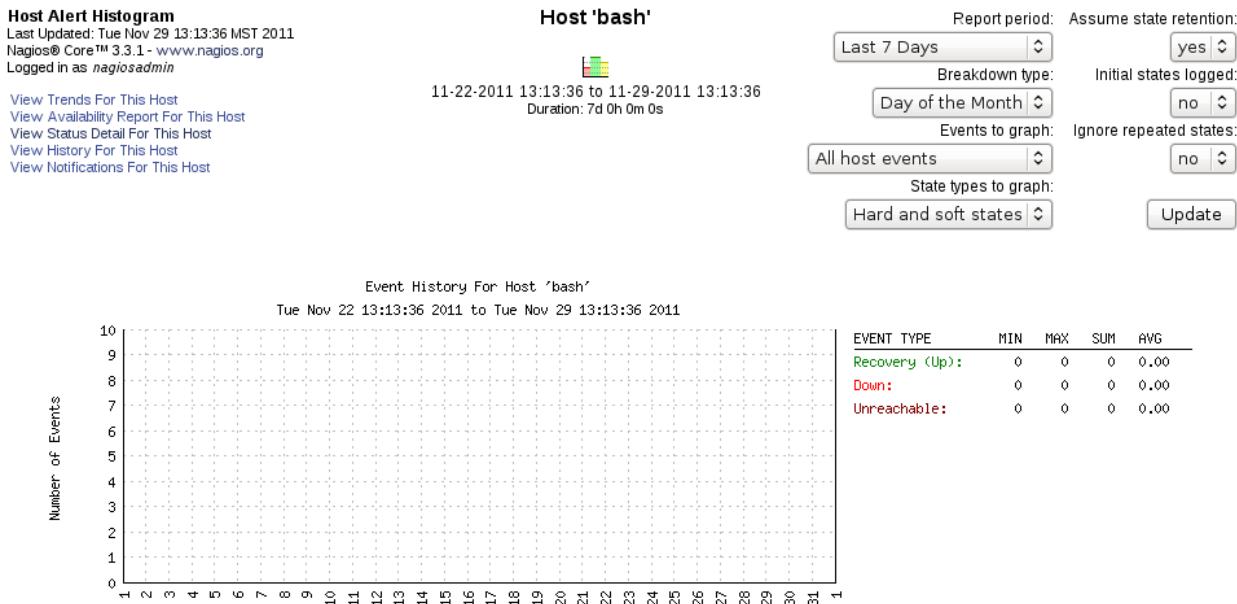
State Types To Graph:

Assume State Retention:

Initial States Logged:

Ignore Repeated States:

The graph has options as well so that you can refresh (update) and see new information. This is the Alert Histogram which could be used for a report about alerts showing the frequency and number of alerts over time.



Notifications

Notifications is simply a list of notifications that have been sent, how they were sent and to whom they were sent. Once again there are options to limit the output.

Contact Notifications
Last Updated: Tue Nov 29 13:30:50 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

All Contacts

Latest Archive Log File Navigation
Tue Nov 29 00:00:00 MST 2011 to Present..
File: /usr/local/nagios/var/nagios.log

Notification detail level for all contacts:
All notifications Older Entries First
Update

Host	Service	Type	Time	Contact	Notification Command	Information
winserver	N/A	HOST DOWN	11-29-2011 13:30:08	nagiosadmin	notify-host-by-email	CRITICAL - Host Unreachable (192.168.5.14)
winserver	N/A	HOST DOWN	11-29-2011 13:30:08	sue	notify-host-by-email	CRITICAL - Host Unreachable (192.168.5.14)
winserver	N/A	HOST DOWN	11-29-2011 13:30:08	john	notify-host-by-email	CRITICAL - Host Unreachable (192.168.5.14)
mail	Updates	WARNING	11-29-2011 13:26:08	nagiosadmin	notify-service-by-email	YUM WARNING: O/S requires an update.

Event Log

The event log represents the information gathered in the Nagios log. The event log provides a way for an administrator to see a log of all that has happened over a period of time. You do have the option to reverse the order seeing older first.

Current Event Log
Last Updated: Tue Nov 29 13:33:39 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as nagiosadmin

Log File Navigation
Tue Nov 29 00:00:00 MST 2011
to Present..

File: /usr/local/nagios/var/nagios.log

November 29, 2011 13:00

```
[11-29-2011 13:30:08] HOST NOTIFICATION: nagiosadmin;winserver;DOWN;notify-host-by-email;CRITICAL - Host Unreachable (192.168.5.14)
[11-29-2011 13:30:08] HOST NOTIFICATION: sue;winserver;DOWN;notify-host-by-email;CRITICAL - Host Unreachable (192.168.5.14)
[11-29-2011 13:30:08] HOST NOTIFICATION: john;winserver;DOWN;notify-host-by-email;CRITICAL - Host Unreachable (192.168.5.14)
[11-29-2011 13:26:08] SERVICE NOTIFICATION: nagiosadmin;mail;Updates;WARNING;notify-service-by-email;YUM WARNING: O/S requires an update.
[11-29-2011 13:26:08] SERVICE NOTIFICATION: sue;mail;Updates;WARNING;notify-service-by-email;YUM WARNING: O/S requires an update.
[11-29-2011 13:26:08] SERVICE NOTIFICATION: john;mail;Updates;WARNING;notify-service-by-email;YUM WARNING: O/S requires an update.
```

Comments

“Comments” allows administrators to communicate about the status of hosts and services. This has the advantage of demonstrating that specific problems are being handled so there is no duplication of effort.

As an administrator the major interest that you will have with the web interface is the ability to recognize and respond to problems. The quickest access to all of the recognized problems is the “Problems” page. This page provides a summary of all problems related to services that Nagios detects.

If an administrator wants to respond to an outage, the host can be selected and then at the bottom of the page a response option is available.

Host Comments

[Add a new comment](#) [Delete all comments](#)

Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions
This host has no comments associated with it							

Here the administrator can “Add a new comment” so that the next administrator recognizes that this problem is in the process of being resolved.

The administrator can now add a comment to indicate the information that is known about the server.

Command Options

Host Name:	<input type="text" value="class"/>
Persistent:	<input checked="" type="checkbox"/>
Author (Your Name):	<input type="text" value="fred"/>
Comment:	<input type="text" value="This host has been confirmed to be down."/>
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

Once this is entered other administrators will be able to see the situation and not repeat the steps that have already

been taken.

Host Comments								
		Add a new comment		Delete all comments				
Entry Time	Author	Comment	Comment ID	Persistent	Type	Expires	Actions	
03-22-2009 14:47:20	Nagios Admin	This host is confirmed down.	9	Yes	User	N/A		

This way administrators can communicate about the situation.

When an administrator is going to take responsibility to solve the problem they can select the “Acknowledge this problem” option in Service Commands.

When the Commands Options opens you have several options. The “Sticky Acknowledgment” when it is checked, will prevent further notifications if the problem continues. The “Send Notifications” when checked, will notify the other administrators so that they do not take action on something that is already being fixed.

Command Options

Host Name:	<input type="text" value="class"/>
Service:	<input type="text" value="User Load"/>
Sticky Acknowledgement:	<input checked="" type="checkbox"/>
Send Notification:	<input checked="" type="checkbox"/>
Persistent Comment:	<input type="checkbox"/>
Author (Your Name):	<input type="text" value="Nagios Admin"/>
Comment:	<input type="text"/>
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

“Persistent Comment” in Nagios 3 will retain the comment even after a reboot and must be manually unchecked when it is fixed. If you leave it unchecked Nagios will remove the comment when a solution is found.

Downtime

If you are going to work on a server or device and need to schedule downtime so Nagios does not notify administrators. This task can be performed at the web interface. When you select the host or service that will be down you have an option to schedule downtime. When downtime is scheduled Nagios will place a comment in the web interface in order to communicate the fact to all administrators who access the web interface.

There are two types of downtime. **Fixed downtime** allows for an exact start and end time when the host or service

will be unavailable. **Flexible downtime** allows for a start time but an open ended startup time as the exact time cannot be determined based on the nature of the situation.

Triggered downtime is when the downtime of a parent will trigger downtime for all of its children. In other words, the downtime for a switch, will impact all of the devices connected to it.

Scheduling Downtime for a Host

In order to schedule downtime for a host, select host details from the web interface. On the right hand side you will notice the “yellow clocks” permit scheduling for host or services. Select the host option.

The screenshot shows the Nagios web interface for managing a host named 'amail'. The top section displays host information: Last Updated (Thu Jan 13 05:23:08 MST 2011), Updated every 90 seconds, Nagios® Core™ 3.2.3 - www.nagios.org, Logged in as nagiosadmin. Below this are links for status, alert history, trends, alert histogram, availability report, and notifications. The host's IP address is listed as 192.168.5.131. To the right is a 'Host' icon and a 'Member of check_mk' icon. A 'Extra Actions' button is also present.

Host State Information

Host Status:	UP (for 0d 0h 42m 9s)
Status Information:	OK - 192.168.5.131: rta 3.577ms, lost 0%
Performance Data:	rta=3.577ms;200.000;500.000;0; pl=0%;40;80;; rtmax=14.098ms;;;; rtmin=0.626ms;;;;
Current Attempt:	1/1 (HARD state)
Last Check Time:	01-13-2011 05:22:41
Check Type:	ACTIVE
Check Latency / Duration:	0.280 / 0.067 seconds
Next Scheduled Active Check:	01-13-2011 05:23:51
Last State Change:	01-13-2011 04:40:59
Last Notification:	N/A (notification 0)
Is This Host Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	01-13-2011 05:23:01 (0d 0h 0m 7s ago)

Host Commands

Locate host on map
Disable active checks of this host
Re-schedule the next check of this host
Submit passive check result for this host
Stop accepting passive checks for this host
Stop obsessing over this host
Disable notifications for this host
Send custom host notification
Schedule downtime for this host
Schedule downtime for all services on this host
Disable notifications for all services on this host
Enable notifications for all services on this host
Schedule a check of all services on this host
Disable checks of all services on this host
Enable checks of all services on this host
Enable event handler for this host
Disable flap detection for this host

Host Comments

Add a new comment | Delete all comments

Entry Time Author Comment Comment ID Persistent Type Expires Actions

Once you have selected the host, “Command Options” appears and provides a place to explain why the downtime to other administrators in the comment area, which is a good idea in most situations. If you select a “Fixed” time you will enter the start and end of the downtime. If this machine provided network connection with other devices you may want to notify downstream devices with a “triggered by” option that is created by this device going down. Or you may choose to do nothing.

Command Options

Host Name:	amail
Author (Your Name):	Nagios Admin
Comment:	Rebuilding SCSI Disk
Triggered By:	N/A
Start Time:	01-13-2011 05:23:36
End Time:	01-13-2011 07:23:36
Type:	Fixed
If Flexible, Duration:	2 Hours 0 Minutes
Child Hosts:	Do nothing with child hosts
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

On the Nagios interface on the left menu, if you select “Downtime” you will see a list of all scheduled downtimes for hosts and services. Remember it may take a few minutes to allow the devices to show up.

All Host and Service Scheduled Downtime
Last Updated: Thu Jan 13 05:24:28 MST 2011
Updated every 90 seconds
Nagios® Core™ 3.2.3 - www.nagios.org
Logged in as nagiosadmin

[Host Downtime | Service Downtime]

Scheduled Host Downtime
🕒 Schedule host downtime

HostName	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
There are no hosts with scheduled downtime										

Scheduled Service Downtime
🕒 Schedule service downtime

Host Name	Service	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
There are no services with scheduled downtime											

Here is how the host looks with downtime (this is the exfoliation frontend), note the “yellow clock” which is an indicator of scheduled downtime.

amail					UP	01-13-2011 05:23:51	0d 0h 43m 54s	OK - 192.168.5.131: rta 2.269ms, lost 0%
centos					UP	01-13-2011 05:24:41	0d 0h 0m 52s	PING OK - Packet loss = 0%, RTA = 0.70 ms

If you select the clock you will see the details on the host list it as being in a scheduled downtime.

Host State Information	
Host Status:	UP (for 0d 0h 44m 12s)
Status Information:	OK - 192.168.5.131: rta 1.389ms, lost 0%
Performance Data:	rta=1.389ms;200.000;500.000;0; pl=0%; 40;80;; rtmax=4.720ms;;;; rtmin=0.495ms;;;;
Current Attempt:	1/1 (HARD state)
Last Check Time:	01-13-2011 05:25:01
Check Type:	ACTIVE
Check Latency / Duration:	0.208 / 0.046 seconds
Next Scheduled Active Check:	01-13-2011 05:26:11
Last State Change:	01-13-2011 04:40:59
Last Notification:	N/A (notification 0)
Is This Host Flapping?	NO (0.00% state change)
In Scheduled Downtime?	YES
Last Update:	01-13-2011 05:25:11 (0d 0h 0m 0s ago)

At this point it will be listed in the “Downtime” menu. Note you can cancel by deleting the downtime.

[[Host Downtime](#) | [Service Downtime](#)]

Scheduled Host Downtime

 [Schedule host downtime](#)

Host Name	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
amail	01-13-2011 05:24:23	Nagios Admin	Rebuilding SCSI Disk	01-13-2011 05:23:36	01-13-2011 07:23:36	Fixed	0d 2h 0m 0s	1	N/A	

Notifications for downtime should stop in the downtime period. If the notifications do not stop verify that you do not have the “d” option set for your contacts. The “d” option will send notifications on downtime.

Process Info

The Nagios process can be managed from this window. All of the options for stopping/starting/restarting the process are here just as the ability to turn on/off notifications, service checks, passive checks, event handlers, flapping, and performance data.

Nagios Process Information
 Last Updated: Tue Nov 29 13:46:23 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Process Information		Process Commands
Program Version:	3.3.1	<input checked="" type="checkbox"/> Shutdown the Nagios process
Program Start Time:	11-29-2011 06:05:07	<input checked="" type="checkbox"/> Restart the Nagios process
Total Running Time:	0d 7h 41m 16s	<input checked="" type="checkbox"/> Disable notifications
Last External Command Check:	11-29-2011 13:46:17	<input checked="" type="checkbox"/> Stop executing service checks
Last Log File Rotation:	N/A	<input checked="" type="checkbox"/> Stop accepting passive service checks
Nagios PID	21937	<input checked="" type="checkbox"/> Stop executing host checks
Notifications Enabled?	YES	<input checked="" type="checkbox"/> Stop accepting passive host checks
Service Checks Being Executed?	YES	<input checked="" type="checkbox"/> Disable event handlers
Passive Service Checks Being Accepted?	YES	<input checked="" type="checkbox"/> Start obsessing over services
Host Checks Being Executed?	YES	<input checked="" type="checkbox"/> Start obsessing over hosts
Passive Host Checks Being Accepted?	YES	<input checked="" type="checkbox"/> Disable flap detection
Event Handlers Enabled?	Yes	<input checked="" type="checkbox"/> Disable performance data
Obsessing Over Services?	No	
Obsessing Over Hosts?	No	
Flap Detection Enabled?	Yes	
Performance Data Being Processed?	Yes	

Performance Info

Performance information can be viewed in a summary format by going to “System” “Performance Info” in the menu. This summary gives you a idea about what is occurring on the Nagios server. This chart will list the services and hosts and break up the checks based on whether they are active or passive checks and also list the latency information based on those checks. The “Check Statistics” provides a summary of checks over three different time periods, 1,5, and 15 minute intervals.

All of this information is based on the nagiostats command which can provides the same information at the command line:

nagiostats

or use the full path to the command

/usr/local/nagios/bin/nagiostats

Performance Information

Last Updated: Thu Mar 22 09:46:04 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Program-Wide Performance Information																																																					
Services Actively Checked:		<table border="1"> <thead> <tr> <th>Time Frame</th><th>Services Checked</th><th>Min.</th><th>Max.</th><th>Average</th></tr> </thead> <tbody> <tr> <td><= 1 minute:</td><td>3 (2.5%)</td><td>0.00 sec</td><td>10.19 sec</td><td>1.698 sec</td></tr> <tr> <td><= 5 minutes:</td><td>51 (43.2%)</td><td>0.00 sec</td><td>0.31 sec</td><td>0.122 sec</td></tr> <tr> <td><= 15 minutes:</td><td>115 (97.5%)</td><td></td><td></td><td></td></tr> <tr> <td><= 1 hour:</td><td>115 (97.5%)</td><td></td><td></td><td></td></tr> <tr> <td>Since program start:</td><td>115 (97.5%)</td><td></td><td></td><td></td></tr> </tbody> </table>				Time Frame	Services Checked	Min.	Max.	Average	<= 1 minute:	3 (2.5%)	0.00 sec	10.19 sec	1.698 sec	<= 5 minutes:	51 (43.2%)	0.00 sec	0.31 sec	0.122 sec	<= 15 minutes:	115 (97.5%)				<= 1 hour:	115 (97.5%)				Since program start:	115 (97.5%)																					
Time Frame	Services Checked	Min.	Max.	Average																																																	
<= 1 minute:	3 (2.5%)	0.00 sec	10.19 sec	1.698 sec																																																	
<= 5 minutes:	51 (43.2%)	0.00 sec	0.31 sec	0.122 sec																																																	
<= 15 minutes:	115 (97.5%)																																																				
<= 1 hour:	115 (97.5%)																																																				
Since program start:	115 (97.5%)																																																				
Services Passively Checked:		<table border="1"> <thead> <tr> <th>Metric</th><th>Min.</th><th>Max.</th><th>Average</th></tr> </thead> <tbody> <tr> <td>Percent State Change:</td><td>0.00%</td><td>30.00%</td><td>0.25%</td></tr> </tbody> </table>				Metric	Min.	Max.	Average	Percent State Change:	0.00%	30.00%	0.25%																																								
Metric	Min.	Max.	Average																																																		
Percent State Change:	0.00%	30.00%	0.25%																																																		
Hosts Actively Checked:		<table border="1"> <thead> <tr> <th>Metric</th><th>Min.</th><th>Max.</th><th>Average</th></tr> </thead> <tbody> <tr> <td>Check Execution Time:</td><td>0.00 sec</td><td>4.07 sec</td><td>3.011 sec</td></tr> <tr> <td>Check Latency:</td><td>0.00 sec</td><td>0.35 sec</td><td>0.216 sec</td></tr> <tr> <td>Percent State Change:</td><td>0.00%</td><td>0.00%</td><td>0.00%</td></tr> </tbody> </table>				Metric	Min.	Max.	Average	Check Execution Time:	0.00 sec	4.07 sec	3.011 sec	Check Latency:	0.00 sec	0.35 sec	0.216 sec	Percent State Change:	0.00%	0.00%	0.00%																																
Metric	Min.	Max.	Average																																																		
Check Execution Time:	0.00 sec	4.07 sec	3.011 sec																																																		
Check Latency:	0.00 sec	0.35 sec	0.216 sec																																																		
Percent State Change:	0.00%	0.00%	0.00%																																																		
Hosts Passively Checked:		<table border="1"> <thead> <tr> <th>Metric</th><th>Min.</th><th>Max.</th><th>Average</th></tr> </thead> <tbody> <tr> <td>Percent State Change:</td><td>0.00%</td><td>0.00%</td><td>0.00%</td></tr> </tbody> </table>				Metric	Min.	Max.	Average	Percent State Change:	0.00%	0.00%	0.00%																																								
Metric	Min.	Max.	Average																																																		
Percent State Change:	0.00%	0.00%	0.00%																																																		
Check Statistics:		<table border="1"> <thead> <tr> <th>Type</th><th>Last 1 Min</th><th>Last 5 Min</th><th>Last 15 Min</th></tr> </thead> <tbody> <tr> <td>Active Scheduled Host Checks</td><td>2</td><td>14</td><td>44</td></tr> <tr> <td>Active On-Demand Host Checks</td><td>3</td><td>11</td><td>44</td></tr> <tr> <td>Parallel Host Checks</td><td>3</td><td>16</td><td>49</td></tr> <tr> <td>Serial Host Checks</td><td>0</td><td>0</td><td>0</td></tr> <tr> <td>Cached Host Checks</td><td>2</td><td>9</td><td>39</td></tr> <tr> <td>Passive Host Checks</td><td>0</td><td>3</td><td>8</td></tr> <tr> <td>Active Scheduled Service Checks</td><td>3</td><td>61</td><td>197</td></tr> <tr> <td>Active On-Demand Service Checks</td><td>0</td><td>0</td><td>0</td></tr> <tr> <td>Cached Service Checks</td><td>0</td><td>0</td><td>0</td></tr> <tr> <td>Passive Service Checks</td><td>4</td><td>13</td><td>38</td></tr> <tr> <td>External Commands</td><td>4</td><td>17</td><td>49</td></tr> </tbody> </table>				Type	Last 1 Min	Last 5 Min	Last 15 Min	Active Scheduled Host Checks	2	14	44	Active On-Demand Host Checks	3	11	44	Parallel Host Checks	3	16	49	Serial Host Checks	0	0	0	Cached Host Checks	2	9	39	Passive Host Checks	0	3	8	Active Scheduled Service Checks	3	61	197	Active On-Demand Service Checks	0	0	0	Cached Service Checks	0	0	0	Passive Service Checks	4	13	38	External Commands	4	17	49
Type	Last 1 Min	Last 5 Min	Last 15 Min																																																		
Active Scheduled Host Checks	2	14	44																																																		
Active On-Demand Host Checks	3	11	44																																																		
Parallel Host Checks	3	16	49																																																		
Serial Host Checks	0	0	0																																																		
Cached Host Checks	2	9	39																																																		
Passive Host Checks	0	3	8																																																		
Active Scheduled Service Checks	3	61	197																																																		
Active On-Demand Service Checks	0	0	0																																																		
Cached Service Checks	0	0	0																																																		
Passive Service Checks	4	13	38																																																		
External Commands	4	17	49																																																		
Buffer Usage:		<table border="1"> <thead> <tr> <th>Type</th><th>In Use</th><th>Max Used</th><th>Total Available</th></tr> </thead> <tbody> <tr> <td>External Commands</td><td>0</td><td>9</td><td>4096</td></tr> </tbody> </table>				Type	In Use	Max Used	Total Available	External Commands	0	9	4096																																								
Type	In Use	Max Used	Total Available																																																		
External Commands	0	9	4096																																																		

Scheduling Queue

The scheduling queue gives you a window into how Nagios is performing by being able to see what is coming up in the queue. This will help you understand queue health and detect latency issues.

Check Scheduling Queue

Last Updated: Tue Nov 29 13:52:56 MST 2011
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Entries sorted by **next check time** (ascending)

Host ↑↓	Service ↑↓	Last Check ↑↓	Next Check ↑↓	Type	Active Checks	Actions
ub		11-29-2011 13:47:38	11-29-2011 13:52:48	Normal	ENABLED	X (L)
bash	NTP	11-29-2011 13:42:48	11-29-2011 13:52:48	Normal	ENABLED	X (L)
localhost	SSH	11-29-2011 13:47:55	11-29-2011 13:52:55	Normal	ENABLED	X (L)
localhost	HTTP	11-29-2011 13:47:55	11-29-2011 13:52:55	Normal	ENABLED	X (L)
bash	SSH Check_Multi	11-29-2011 13:42:59	11-29-2011 13:52:59	Normal	ENABLED	X (L)
mail	Processes	11-29-2011 13:43:04	11-29-2011 13:53:04	Normal	ENABLED	X (L)

By selecting the clock you can alter the schedule for a service check. Once you enter “Commit” the queue will change.

External Command Interface

Last Updated: Tue Nov 29 13:55:04 MST 2011
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

You are requesting to schedule a service check

Command Options	Command Description
Host Name: <input type="text" value="mail"/> Service: <input type="text" value="Quarantine Status"/> Check Time: <input type="text" value="11-29-2011 13:55:04"/> Force Check: <input checked="" type="checkbox"/>	Command Description <div style="border: 1px solid #ccc; padding: 5px;"> This command is used to schedule the next check of a particular service. Nagios will re-queue the service to be checked at the time you specify. If you select the <i>force check</i> option, Nagios will force a check of the service regardless of both what time the scheduled check occurs and whether or not checks are enabled for the service. </div>
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

Configuration

This link give you the option to modify the configuration of your hosts and services from the web interface, not the command line. As you can see in the image this scrolls off one page into numerous pages of options.

Configuration
Last Updated: Tue Nov 29 13:58:00 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Object Type:

Show Only Services Named or on Host:

Services

Service		Max. Check Attempts	Normal Check Interval	Retry Check Interval	Check Command	Check Period	Parallelize	Volatile	Obsess Over	Enable Active Checks	Enable Passive Checks	Check Freshness	Freshness Threshold	Def. C.
bash	Check Postfix	3	0h 10m 0s	0h 2m 0s	check_nrpe!check_postfix	24x7	Yes	No	Yes	Yes	Yes	No	Auto-determined value	edit
bash	DNS	3	0h 10m 0s	0h 2m 0s	check_nrpe!check_dns	24x7	Yes	No	Yes	Yes	Yes	No	Auto-determined value	edit
bash	Disk sda1	3	0h 10m 0s	0h 2m 0s	check_nrpe!check_sda1	24x7	Yes	No	Yes	Yes	Yes	No	Auto-determined value	edit
bash	Load	3	0h 10m 0s	0h 2m 0s	check_nrpe!check_load	24x7	Yes	No	Yes	Yes	Yes	No	Auto-determined value	edit

If you select a link you will be able to modify that aspect of the service or host check. Here is an example of a “Check Command” modification. Just make the changes and select “Update”.

Configuration
Last Updated: Tue Nov 29 14:04:51 MST 2011
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Object Type:

Command Expansion

Command Name	Command Line
To expand:	<code>check_nrpe!check_dns</code>
<code>check_nrpe</code>	<code>\$USER1\$!check_nrpe -H \$HOSTADDRESS\$ -c \$ARG1\$</code>
->	<code>\$USER1\$!check_nrpe -H \$HOSTADDRESS\$ -c check_dns</code>
To expand:	<code>check_nrpe!check_dns</code>

Event Handlers

Event handlers are options to use when the host or service changes between an OK state to an error state. An administrator may implement a “self-healing” script which will repair a situation before anyone is notified. Now “self-healing” of course is a stretch because situations that arise repeatedly need to be examined by an administrator and fixed properly. And yet even handlers do have a place in maintaining an organizations hosts and services.

There are several handler types that may be implemented; global service and host event handlers and host and service specific event handlers. If global event handlers are run of course they will run for all hosts and services. Typically organizations will select specific hosts or services to run event handlers on.

Event Handler for Nagios Server

This example of setting up an event handler is performed on the localhost, or Nagios server. The goal is to restart the web interface if it fails. There are four elements to setting up an event handler; the service definition, the command definition; the script for the event handler and the permissions required.

The service is the typical check_http service definition but an additional line has been added for the event_handler. This file is the localhost.cfg file and the service shows that this is a service-specific event handler. The event handler name must match that of the command definition.

```
define service{
    use                               local-service
    host_name                         localhost
    service_description                HTTP
    check_command                     check_http
    event_handler                     httpd-restart
}
```

The commands.cfg contains definitions of commands and is where the definition for the event handler must be entered. This is an event handler for a service so these macros must be added after the script name:

`$SERVICESTATE$ $SERVICESTATETYPE$ $SERVICEATTEMPT$` . If it was a host these macros would be required: `$HOSTSTATE$ $HOSTSTATETYPE$ $HOSTATTEMPT$` . Note the location and name of the event handler script. Your script name will vary.

```
define command{
    command_name        httpd-restart
    command_line        $USER1$/eventhandlers/httpd-restart.sh $SERVICESTATE$
$SERVICESTATETYPE$ $SERVICEATTEMPT$
}
```

The event handler script in this example will attempt to restart the web server after 3 SOFT problem states and once again after the HARD state is reached if it has not been restarted. Paths for the commands indicated may change based on the Linux distro used so check paths with:

```
which sudo
which service
```

```
#!/bin/sh
# Event Handler for Web Server on Nagios

case "$1" in
OK)
;;
WARNING)
;;
UNKNOWN)
;;
CRITICAL)
    case "$2" in
```

```

SOFT)
    case "$3" in
        3)
            echo -n "Restarting Web service"
            /usr/bin/sudo /sbin/service httpd restart
            ;;
            esac
        ;;
HARD)
    echo -n "Restarting Web service"
    /usr/bin/sudo /sbin/service httpd restart
    ;;
esac
;;
esac
exit 0

```

Once the file has been saved change the permissions and ownership so that it is executable and is owned by nagios.

```

chmod 755 httpd-restart.sh
chown nagios http-restart.sh

```

The permissions to execute a service will need to be modified.



Security Tip

Whenever sudo is used it is important to consider the security implications. In this example the nagios user is able to elevate privileges to root in order to execute the restart of a service. Note that the nagios user is not required to have a password to obtain these rights. Only root can restart services, this is why it is required. Open visudo as root and add these lines.

```

User_Alias NAGIOS = nagios,nagiocmd
Cmnd_Alias NAGIOSCOM = /sbin/service, /etc/rc.d/init.d/httpd
Defaults:NAGIOS !requiretty
NAGIOS      ALL=(ALL)      NOPASSWD: NAGIOSCOM

```

Once you have saved changes now test the set up by turning the web server off and viewing logs.

```
service httpd stop
```

Here is an example of the log file output indicating that the httpd server is up (HTTP;OK;SOFT) and then showing 3 SOFT problem states after which the script executes and the web server is running again.

```
tail /var/log/nagios/nagios.log
Nov 21 06:59:18 nag2 nagios: SERVICE EVENT HANDLER: localhost;HTTP;OK;SOFT;4;httpd-restart
Nov 21 07:04:13 nag2 nagios: SERVICE EVENT HANDLER: localhost;HTTP;CRITICAL;SOFT;1;httpd-
restart
Nov 21 07:05:16 nag2 nagios: SERVICE EVENT HANDLER: localhost;HTTP;CRITICAL;SOFT;2;httpd-
restart
Nov 21 07:06:22 nag2 nagios: SERVICE EVENT HANDLER: localhost;HTTP;CRITICAL;SOFT;3;httpd-
restart
        service httpd status
httpd (pid 25826) is running...
```

Host Groups

Often you will want to create a group of devices that have similar monitoring needs. The hostgroup allows you to then create service checks that monitor all of the devices in the hostgroup. Specifically what this means is that the services defined for the group will be available for all hosts in the group without making individual configurations. Nagios will also list the hosts together in the web interface if they are in the same hostgroup.

Define Each Host

In order to set up a hostgroup, each server must be defined as a host. In this example, 3 Ubuntu servers are defined.

```
define host{
    use                  linux-server
    host_name           ub
    alias               Ubuntu Server
    address             192.168.5.180
}
define host{
    use                  linux-server
    host_name           ub1
    alias               Ubuntu Server
    address             192.168.5.181
}
define host{
    use                  linux-server
    host_name           ub3
    alias               Ubuntu Server
    address             192.168.5.183
}
```

Define Host Groups

Create hostgroups.cfg in the objects directory and create an entry in nagios.cfg to the location of hostgroups.cfg.

```
cfg_file=/usr/local/nagios/etc/objects/hostgroup.cfg
```

Define the hostgroup, in this example the hostgroup `ubuntu_servers` is defined with the three members that were defined in hosts.cfg file.

```
define hostgroup {
    hostgroup_name      ubuntu_servers
    alias               Ubuntu Servers
    members             ub,ub1,ub3
}
```

Define Services for the Group

The advantage of the hostgroup is that you can create one service definition and add that to the whole group of servers. This is exactly the same as a regular service definition except you use hostgroup_name instead of host.

```
define service{
    use                  generic-service
    hostgroup_name       ubuntu_servers
    service_description  Ping
    check_command        check_ping!60.0,5%!100.0,10%
}
define service{
    use                  generic-service
    hostgroup_name       ubuntu_servers
    service_description  SSH Server
    check_command        check_tcp!22
}
define service{
    use                  generic-service
    hostgroup_name       ubuntu_servers
    service_description  Web Server
    check_command        check_tcp!80
}
```

Now if you go to the web interface and select “Hostgroups” you will have a group of servers that are all related with the same service checks.

Current Network Status
Last Updated: Sat Jan 29 10:26:15 MST 2011
Updated every 90 seconds
Nagios® Core™ 3.2.3 - www.nagios.org
Logged in as **nagiosadmin**

[View Service Status Detail For All Host Groups](#)
[View Host Status Detail For This Host Group](#)
[View Status Overview For This Host Group](#)
[View Status Summary For This Host Group](#)
[View Status Grid For This Host Group](#)



Service Status Details For Host Group 'ubuntu_servers'

Host ↑↓	Service ↑↓	Status ↑↓	Last Check ↑↓	Duration ↑↓	Attempt ↑↓	Status Information
ub	Ping	OK	01-29-2011 10:22:32	0d 0h 23m 43s	1/3	PING OK - Packet loss = 0%, RTA = 1.42 ms
	SSH Server	OK	01-29-2011 10:20:18	0d 0h 15m 57s	1/3	TCP OK - 0.001 second response time on port 22
	Web Server	OK	01-29-2011 10:21:01	0d 0h 5m 14s	1/3	TCP OK - 0.082 second response time on port 80
ub1	Ping	OK	01-29-2011 10:25:22	0d 0h 20m 53s	1/3	PING OK - Packet loss = 0%, RTA = 1.03 ms
	SSH Server	OK	01-29-2011 10:24:45	0d 0h 21m 30s	1/3	TCP OK - 0.681 second response time on port 22
	Web Server	OK	01-29-2011 10:23:14	0d 0h 3m 1s	1/3	TCP OK - 0.008 second response time on port 80
ub3	Ping	OK	01-29-2011 10:18:12	0d 0h 18m 3s	1/3	PING OK - Packet loss = 0%, RTA = 0.71 ms
	SSH Server	OK	01-29-2011 10:17:37	0d 0h 18m 38s	1/3	TCP OK - 0.009 second response time on port 22
	Web Server	OK	01-29-2011 10:25:27	0d 0h 0m 48s	1/3	TCP OK - 0.431 second response time on port 80

If you want to add individual service checks for one of the servers in the hostgroup that would be done as a regular service definition using the host.

Service Groups

Nagios combines devices that are checking the same services into groups in order to make the set up faster and more efficient. This allows an administrator to group machines based on services. Each of these services must be configured as service checks for each host. Once that is complete the services may be grouped in the servicegroups.cfg. The other major advantage is that the administrator may manage all those in the service group with servicegroup commands in the web interface.

You will need to create a file called servicegroups.cfg and put an entry in nagios.cfg to indicate where it is. Note the entries are in pairs (first host, then service) "host,service, host2,service2".

```
define servicegroup{
    servicegroup_name      web
    alias                  Web Servers
    members                ub, HTTP ,ub1, HTTP ,ub3, HTTP
}
```

Define each host with a normal service check.

```
define service{
    use                   generic-service
```

```

host_name          ub
service_description HTTP
check_command      check_http

}

define service{
    use           generic-service
    host_name    ub1
    service_description HTTP
    check_command  check_http
}

define service{
    use           generic-service
    host_name    ub3
    service_description HTTP
    check_command  check_http
}

```

This now allows the administrator to group these services and view them as a group when “ServiceGroups” is selected in the web interface.

Service Overview For All Service Groups

Web Servers (web)			
Host	Status	Services	Actions
ub	UP	1 OK	
ub1	UP	1 OK	
ub3	UP	1 OK	

Managing Nagios Time

The correct time on a server is critical, especially for Nagios. Synchronized time with other servers is important to coordinate tasks and discover problems. One easy way to perform this task is to install ntp (Network Time Protocol) and then manually have the system check time once a day in order to manage time.

```
yum install -y ntp
```

Manually update your time. Here you can see the system is off by 12 seconds so it will be corrected gradually with each check.

```
ntpdate pool.ntp.org
11 Mar 12:09:55 ntpdate[24725]: step time server 71.245.107.83 offset 10900.806712 sec
12 sec
```

Create a cron entry so that each day your system is brought up to date.

```
crontab -e
```

```
45 23 * * * /usr/sbin/ntpdate pool.ntp.org
```

Nagios Core BackUp

Most of the important information that is used for Nagios is located in the /usr/local/nagios directory. However, if you use additional addons this will make it more difficult to find all of the files you need. Here are some common directories that you may want to add:

```
/usr/local/pnp4nagios
/usr/local/nagvis
```

You can also search for all files on the system that are owned by Nagios with:

```
find / -user nagios
```

Nagios Core Backups with Flat Files

Weekly Timestamped Backups

The weekly backups should be placed on a separate disk from the disk that Nagios is on. That will at least give you way to rebuild even if you had to move the disk to a new location. Timestamps are important in that they allow you to return to a known date. Make sure you provide enough disk space so that you can save 6 months worth. The script creates a time stamp so that you know not only when it was created but so that no file will ever be able to overwrite it. The time stamp is year, month, day, hour, minute and second so there will be no two the same. The “echo” command makes sure that on each execution of the script all files in the script have the same time stamp.

bk.sh

```
#!/bin/bash
# Weekly Backup
TIMESTAMP=`date +%Y%m%d_%H%M%S`;
echo $TIMESTAMP
tar -czvf /bk/nagios_$TIMESTAMP.tar.gz /usr/local/nagios
tar -czvf /bk/httpd_$TIMESTAMP.tar.gz /etc/httpd
```

Daily Backups

Daily backups are provided so that you have quick access to restore a day's work. Note, these backups are overwritten each day because the backup name is the same.

daily.sh

```
#!/bin/bash
# Daily Backup
tar -czvf /bk/nagios.tar.gz /usr/local/nagios
tar -czvf /bk/httpd.tar.gz /etc/httpd
```

Backup Directory

Make sure this is a separate drive. You can then copy backups to offsite or another location as well. It is good to maintain these files on the Nagios server so they are handy to get to. The backup directory should have two sets of files, timestamped and non-timestamped.

```
/bk
httpd_20110403_171355.tar.gz
httpd.tar.gz
nagios_20110403_171355.tar.gz
nagios.tar.gz
```

Nagios Core with MySQL Database

You must also perform a proper backup of your MySQL database if you are using MySQL. The user that performs mysqldump must be the root user and you will need to either do this manually or add the password to the script.

Weekly Timestamped Backups with MySQL

bk.sh

```
#!/bin/bash
# Weekly Backup
TIMESTAMP=`date +%Y%m%d_%H%M%S`;
echo $TIMESTAMP

tar -czvf /bk/nagios_$TIMESTAMP.tar.gz /usr/local/nagios
tar -czvf /bk/httpd_$TIMESTAMP.tar.gz /etc/httpd
mysqldump -u root --password=your_password nagios > /bk/nagios_sql_$TIMESTAMP
```

Daily Backups with MySQL

```
daily.sh
#!/bin/bash
# Daily Backup
```

```
tar -czvf /bk/nagios.tar.gz /usr/local/nagios  
tar -czvf /bk/httpd.tar.gz /etc/httpd  
mysqldump -u root --password=your_password nagios > /bk/nagios.sql
```

Restore Backups

If you are going to use backups the actual backup is only half the story. You must practice restoring information that is backed up. When you restore tar files you must tell tar that since the files were created in reference to the “/” directory you need to restore them that was so you will need to add “-C /”.

```
tar -xzvf /bk/nagios.tar.gz -C /  
tar -xzvf /bk/httpd.tar.gz -C /
```

Automatically BackUp

It only makes sense to create cron jobs that make the backup process automatic. The tool to use for cronjobs is crontab. The first thing you need to verify is the location of the scripts you will use. This is especially important with the crontab is that you wan to use the full path for all scripts and commands.

Open a cronjob as root with:

```
crontab -e
```

The “-e” is for edit and it will open an empty file is using CentOS. You will need to edit the file with vi so in order to add text you must use the “i” to enter edit mode.

cron Format

There are six fields that must be used.

- 1 - minute 0-59, a - between numbers means a range 1-30, a comma between numbers means individual 1,5,8
- 2 - hour 0-23
- 3 - day of month 0-31
- 4 - month 0-12
- 5 - day of week 0-7 (both 0 and 7 are Sunday)
- 6 - command

Example: Run a program at 3:14 every day.

```
14 3 * * * /root/scripts./bk.sh
```

Edit the crontab and then save in the normal vi fashion.

ESC -to get out of edit mode.

:wq - to save

List the crontabs that you currently have with:

```
crontab -l
```

It is important that you verify the backups are working and that you are capable of restoring them.

Reachability

Nagios has the ability to determine if a host is in a down state or if it is in an unreachable state. The practical implications of both of these states is the same, stuff does not work. However, the troubleshooting aspect is quite different. If a host is down, then of course the administrator needs to investigate the host specifically. However, if a network device is down or so heavily loaded it restricts communication then the network administrator needs to focus on the network devices and related issues. So reachability is concerned with the overall network health and how it impacts your monitored hosts.

Nagios is able to discern the network structure and how it alters these down states and unreachable states by understanding the path for data packets on the network. In other words, Nagios needs to know how equipment is connected because that will help determine the situation. This is done by making a reference to the parent/child relationships of connected network devices. This process allows Nagios to take into account the physical topology of the network.

The next step in configuration is to look at the IP Address and hostname of the next network device. If Nagios is connected to a switch then that should be also configured with a host definition. The difference is that you want to tell Nagios that the parent of that switch device is the hostname nagios or localhost.

```
define host{
    host_name      ciscoswitch
    parents        localhost
}
```

You can only add devices that have the ability to be assigned an IP Address and /or a hostname.

The key in the design is recognizing the network configuration and telling Nagios which is the parent, or network device, directly above the host you are working with. Once your data packet hits your external interface on your router you cannot specify routers on the Internet as the path will vary depending upon best route. So if you were tracing the data packet path from Nagios to a remote device you would need to indicate the IP Address of the external router connecting the device to the Internet.

Virtual Hosting Example

A good example of a parent/child relationship is when a virtual host is used to manage several containers. The concern is if the host goes down of course all of the containers will go down as well.

```
vzlist -a
CTID  NPROC STATUS   IP_ADDR      HOSTNAME
161    25 running  192.168.5.161  mk
162    22 running  192.168.5.162  nagvis
```

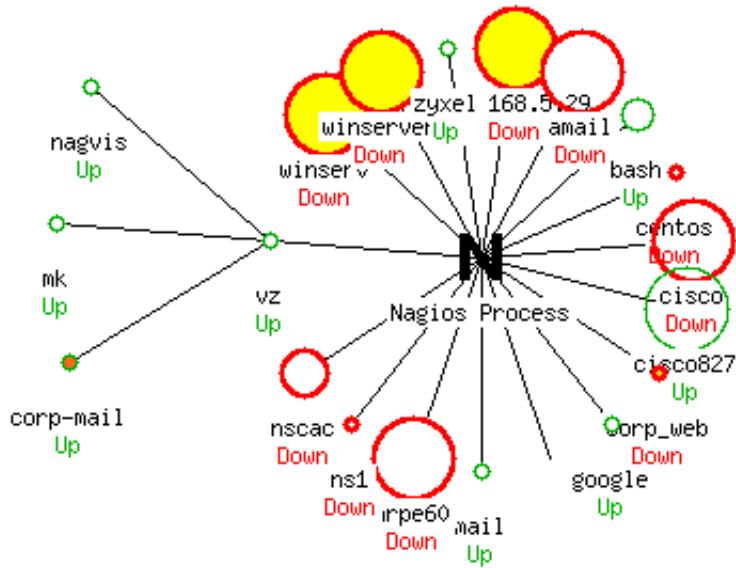
```
170      - stopped 192.168.5.170 host
172      - stopped 192.168.5.172 mail
173      16 running 192.168.5.173 test
174      14 running 192.168.5.174 test2
180      18 running 192.168.5.180 ub
181      9 running 192.168.5.181 ub2
183      9 running 192.168.5.183 ub3
190      17 running 192.168.5.190 bash
```

Here is an example of the host definitions for several containers as well as the parent. Note that vz is listed as “parents” in the host definition.

```
vz (parent)
  nagvis (container)
  corp-mail (container)
  mk (container)

define host{
    use                  linux-server
    host_name           vz
    alias               VZ Server
    address             192.168.5.160
}
define host{
    use                  linux-server
    host_name           nagvis
    alias               Mapping Server
    address             192.168.5.162
    parents             vz
}
define host{
    use                  linux-server
    host_name           corp-mail
    alias               Mail Server
    address             192.168.5.172
    parents             vz
}
define host{
    use                  linux-server
    host_name           mk
    alias               Passive Server
    address             192.168.5.161
    parents             vz
}
```

This image demonstrates the relationship that the containers have to the parent.



An additional container does not list vz as “parents” and therefore is treated differently.

```
define host{
    use                     linux-server
    host_name               bash
    alias                   Bash Scripts
    address                 192.168.5.190
}
```

If an administrator needed to schedule downtime for vz it would certainly impact all of the containers. In this example the “Child Hosts” are triggered with this downtime.

Command Options

Host Name:	vz
Author (Your Name):	Nagios Admin
Comment:	Upgrade Disk Space
Triggered By:	N/A
Start Time:	02-28-2011 19:18:19
End Time:	02-28-2011 21:18:19
Type:	Fixed
If Flexible, Duration:	2 Hours 0 Minutes
Child Hosts:	Schedule triggered downtime for all child hosts
<input type="button" value="Commit"/> <input type="button" value="Reset"/>	

Here the scheduled downtime clearly demonstrates that the containers that have “vz” listed as “parents” are all included. However, the container “bash” is not included in the process as it does not list the child/parent relationship with “vz”.

[[Host Downtime](#) | [Service Downtime](#)]

Scheduled Host Downtime

 [Schedule host downtime](#)

Host Name	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
corp-mail	02-28-2011 19:19:34	Nagios Admin	Upgrade Disk Space	02-28-2011 19:18:19	02-28-2011 21:18:19	Fixed	0d 2h 0m 0s	9	6	
mk	02-28-2011 19:19:34	Nagios Admin	Upgrade Disk Space	02-28-2011 19:18:19	02-28-2011 21:18:19	Fixed	0d 2h 0m 0s	8	6	
nagvis	02-28-2011 19:19:34	Nagios Admin	Upgrade Disk Space	02-28-2011 19:18:19	02-28-2011 21:18:19	Fixed	0d 2h 0m 0s	7	6	
vz	02-28-2011 19:19:34	Nagios Admin	Upgrade Disk Space	02-28-2011 19:18:19	02-28-2011 21:18:19	Fixed	0d 2h 0m 0s	6	N/A	

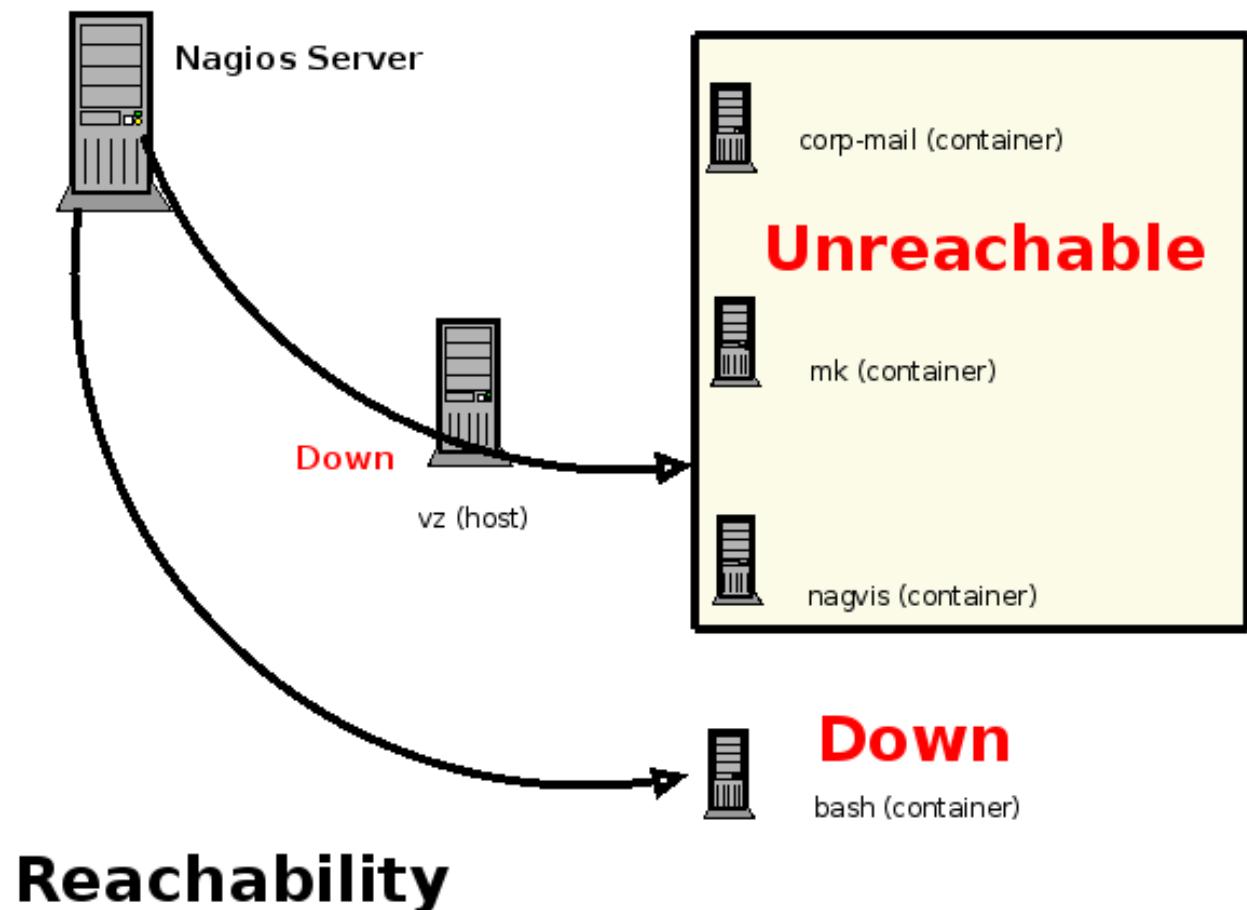
There is another issue here to consider. As an administrator you understand the topology of the network, it is obvious if “vz” is down the containers will also be down. By default the administrator will not only get a notification that “vz” is down but also get notifications that all containers are down as well, possibly unwanted notifications. However, as an administrator you **WILL** want notification on a container that is down but the parent is up.

The default notification options in the linux-server template include notifications for down, unknown and recovery.

```
notification_options      d,u,r
```

The unknown state relates to the example below where Nagios knows the host is down but does not know the state of the containers because they are “unreachable” or “unknown”. Therefore if you did not want notifications for the containers being down, remove the “u” option from notification_options as in the example.

```
define host{
    use                      linux-server
    host_name                nagvis
    alias                     Mapping Server
    address                  192.168.5.162
    parents                   vz
    notification_options     d,r
}
```



If the administrator does not enter the parent/child relationship for “bash” which is a container, then there is not going to be an opportunity to disable notifications for the “unreachable” state as Nagios will believe it is a direct connection and as a result will determine the state to be “down” instead of “unreachable”. Obviously, creating the relationships provides better control of accurate notifications and limiting notifications.

Network Outages

Network outages represent hosts that are in a DOWN state and as a result are blocking downstream hosts and services. Network outages therefore, relate to the parent/child relationship that has been created.

Volatile Service

A volatile service is a service that will automatically return itself to an "OK" status when it is checked. Or, it is a service that needs to be checked by an administrator on each occurrence, like a security event. Volatile services are different than normal services in that:

- * the non-OK state is logged
- * contacts will be notified on each event
- * event handlers are run on each event

Since the state is returned to an OK state after each event, one of the changes that should be made with a volatile service is that the “max_check_attempts” are set to one so that each event will trigger a hard state. If this was set to a higher number than one it would never reach the HARD state as it is reset.

```
is_volatile = 1  
max_check_attempts =1
```

State Stalking

State stalking provides for detailed logging information. The goal in using state stalking is that as much detail as possible is placed in the logs for further review after the event. Instead of just logging state changes, from OK to WARNING for example, stalking logs any changes from the previous check. So not only state changes are logged but information that occurs during the same state is also logged.

```
stalking_options = [o,d,u]
```

The stalking directive provides three options: “o” for stalking on UP states, “d” for stalking on down states and “u” for stalking on UNREACHABLE states.

Flapping

A flapping state is when a service or host changes from an OK state to CRITICAL state rapidly. These changing states will send multitudes of notifications to administrators which can be non-productive. When flapping is detected

Nagios will recognize the changing states and move into a state of flapping which provides additional options for an administrator which could allow unwanted notifications.

In order to detect this flapping state Nagios saves in memory 21 checks for each host and service. Nagios reviews the last 20 changes to determine if the host or service is changing states based on a percentage. In this review of states the more recent checks are provided a greater weight than the older checks as this is probably more important to an administrator. Nagios also provides two thresholds for a service and a host so that an administrator can set an upper and lower threshold which means that when the service or host goes above the upper threshold Nagios recognizes this as state flapping which means notifications will be stopped, an entry in the log is created and a comment is placed in the web interface so it can be reviewed by administrators. Once the percentage goes below the lower limit the comment is removed and the service is returned to a normal state with notifications enabled. This process takes a period of time to occur. Here is an example of a service that is flapping. If you look closely you can see the percentage of state change.

Service State Information	
Current Status:	OK (for 0d 1h 32m 20s)
Status Information:	Explorer.EXE: Running
Performance Data:	
Current Attempt:	1/3 (HARD state)
Last Check Time:	11-02-2010 12:53:19
Check Type:	ACTIVE
Check Latency / Duration:	0.072 / 0.299 seconds
Next Scheduled Check:	11-02-2010 13:03:19
Last State Change:	11-02-2010 11:23:10
Last Notification:	N/A (notification 0)
Is This Service Flapping?	YES (12.70% state change)
In Scheduled Downtime?	NO
Last Update:	11-02-2010 12:55:25 (0d 0h 0m 5s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Notifications for this service are being suppressed because it was detected as having been flapping between different states (12.7% change). When the service state stabilizes and the flapping stops, notifications will be re-enabled.

To make changes to the settings for flap detection, first access the nagios.cfg file which provides global settings. The first setting that can be altered is that an administrator can turn flapping off by changing the value to “0”. The thresholds may be modified to meet specific requirements for the organization. Remember these thresholds are percentages so the low end is 5%, or one state change and the upper end is 20% which equals five state changes.

```
enable_flap_detection=1
```

```
low_service_flap_threshold=5.0
high_service_flap_threshold=20.0
low_host_flap_threshold=5.0
high_host_flap_threshold=20.0
```

Specific changes could be made with the specific service as well. The “flap_detection_enabled” must be included to allow the override of the global settings. The two thresholds then may be modified to meet the needs of the service.

```
define service{
    use generic-service
    host_name centos
    service_description SMTP
    check_command check_smtp
    flap_detection_enabled 1
    low_flap_threshold 10.0
    high_flap_threshold 30.0
}
```

There is another option that is available with flapping. This option allows an administrator to control which states indicated flapping. The states available are o(OK), w(WARNING), c(CRITICAL) and u(UNKNOWN). States that are not listed are not taken into account to determine flapping.

flap_detection_options	o,w,c,u
------------------------	---------

Host State Information

Host Status:	UP (for 0d 0h 1m 35s)
Status Information:	Host Status
Performance Data:	
Current Attempt:	1/10 (HARD state)
Last Check Time:	12-30-2010 00:47:21
Check Type:	PASSIVE
Check Latency / Duration:	N/A / 0.000 seconds
Next Scheduled Active Check:	12-30-2010 00:51:16
Last State Change:	12-30-2010 00:47:26
Last Notification:	N/A (notification 0)
Is This Host Flapping?	YES (24.21% state change)
In Scheduled Downtime?	NO
Last Update:	12-30-2010 00:48:56 (0d 0h 0m 5s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host Commands

■ Locate host on map
X Disable active checks of this host
L Re-schedule the next check of this host
?* Submit passive check result for this host
X Stop accepting passive checks for this host
X Stop obsessing over this host
X Disable notifications for this host
C Send custom host notification
L Schedule downtime for this host
L Schedule downtime for all services on this host
X Disable notifications for all services on this host
✓ Enable notifications for all services on this host
L Schedule a check of all services on this host
X Disable checks of all services on this host
✓ Enable checks of all services on this host
X Disable event handler for this host
X Disable flap detection for this host

Host Comments

+/- Add a new comment X Delete all comments

Entry Time	Author	Comment	
12-30-2010 00:47:26	(Nagios Process)	Notifications for this host are being suppressed because it was detected as having been flapping between different states (24.2% change > 20.0% threshold). When the host state stabilizes and the flapping stops, notifications will be re-enabled.	3 No Flap Detection N/A X

As you can see in this illustration you can also “Disable flap detection for this host” under the “Host Commands”. This provides the option to just perform the task as it happens. Here is the verification before you commit the change.

You are requesting to disable flap detection for a particular host

Command Options		Command Description								
Host Name:	nscac	This command is used to disable flap detection for a specific host.								
		<input type="button" value="Commit"/> <input type="button" value="Reset"/>								
<small>Please enter all required information before committing the command. Required fields are marked in red. Failure to supply all required values will result in an error.</small>										
[Host Downtime Service Downtime]										
Scheduled Host Downtime										
Schedule host downtime										
Host Name	Entry Time	Author	Comment	Start Time	End Time	Type	Duration	Downtime ID	Trigger ID	Actions
amail	01-13-2011 05:24:23	Nagios Admin	Rebuilding SCSI Disk	01-13-2011 05:23:36	01-13-2011 07:23:36	Fixed	0d 2h 0m 0s	1	N/A	

Resolving Problems

If you believe the problem for a host or service has been resolved and you do not want to wait until the check is performed again, you can proceed to the host or service and then click the “Re-schedule the next check” option to get an immediate check performed.

Host Information

Last Updated: Sat Mar 17 08:34:04 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

[View Status Detail For This Host](#)
[View Alert History For This Host](#)
[View Trends For This Host](#)
[View Alert Histogram For This Host](#)
[View Availability Report For This Host](#)
[View Notifications For This Host](#)

Host
My Windows Server
(winserver)

Member of
windows-servers

192.168.5.14

Host State Information	
Host Status:	UP (for 0d 0h 42m 35s)
Status Information:	PING OK - Packet loss = 0%, RTA = 122.57 ms
Performance Data:	rta=122.567001ms;3000.000000;5000.000000;0.000000 pl=0%;80;100;0
Current Attempt:	1/10 (HARD state)
Last Check Time:	03-17-2012 08:32:29
Check Type:	ACTIVE
Check Latency / Duration:	0.204 / 4.010 seconds
Next Scheduled Active Check:	03-17-2012 08:37:39
Last State Change:	03-17-2012 07:51:29
Last Notification:	03-17-2012 07:51:29 (notification 0)
Is This Host Flapping?	NO (5.20% state change)
In Scheduled Downtime?	NO
Last Update:	03-17-2012 08:33:59 (0d 0h 0m 5s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Host Commands

- [Locate host on map](#)
- Disable active checks of this host**
- [Re-schedule the next check of this host](#)
- [Submit passive check result for this host](#)
- [Stop accepting passive checks for this host](#)
- [Stop obsessing over this host](#)
- [Disable notifications for this host](#)
- [Send custom host notification](#)
- [Schedule downtime for this host](#)
- [Schedule downtime for all services on this host](#)
- [Disable notifications for all services on this host](#)
- [Enable notifications for all services on this host](#)
- [Schedule a check of all services on this host](#)
- [Disable checks of all services on this host](#)
- [Enable checks of all services on this host](#)
- [Disable event handler for this host](#)
- [Disable flap detection for this host](#)

Host Comments

[Add a new comment](#) [Delete all comments](#)

Entry Time	Author	Comment	CommentID	Persistent	Type	Expires	Actions
This host has no comments associated with it							

Disabling Notifications

A permanent solution for false notifications is the option to disable notifications for a host or service. Select the host or service and then click “Disable notifications for this service/host”.

Service Information

Last Updated: Sat Mar 17 08:47:55 MDT 2012
Updated every 90 seconds
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Service
AIDE
On Host
Company Server
(**bash**)

Member of
No servicegroups.

192.168.5.190

Service State Information	
Current Status:	CRITICAL (for 43d 17h 32m 52s) (Has been acknowledged)
Status Information:	added: /etc/rc.d/rc3.d/S65dovecot added: /etc/httpd/sales changed: /etc/profile.d changed: /etc/profile.d/krb5-devel.sh changed: /etc/profile.d/krb5-devel.csh changed: /etc/php.ini changed: /etc/postfix changed: /etc/postfix/main.cf changed: /etc/hosts changed: /etc/prelink.cache changed: /etc/rc.d/rc1.d changed: /etc/rc.d/rc1.d/S26lvm2-monitor changed: /etc/rc.d/rc1.d/K35dovecot changed: /etc/rc.d/init.d changed: /etc/rc.d/init.d/lvm2-monitor

Service Commands

- Disable active checks of this service
- Re-schedule the next check of this service
- Submit passive check result for this service
- Stop accepting passive checks for this service
- Stop obsessing over this service
- Remove problem acknowledgement
- Disable notifications for this service
- Delay next service notification
- Send custom service notification
- Schedule downtime for this service
- Disable event handler for this service
- Disable flap detection for this service

Sending Mail From Nagios

Every Linux server that is installed, uses a mail server to send mail locally. The mail server, whether it is Sendmail or Postfix, is enabled to send mail by default but not to receive mail. In addition, the mail server is configured to run at start up of the server. You can confirm this with the netstat command:

```
netstat -aunt
tcp      0      0 127.0.0.1:25          0.0.0.0:*
                                         LISTEN
```

Here the fact the mail server is running on port 25 and is listening, but only on the localhost, 127.0.0.1. What this means is that the mail server will be sending reports to the root user on the localhost and could send mail to another mail server on port 25, but it could not receive any mail outside of the mail server. That is good news, you do not want to turn your Nagios server into a mail relay. Nagios is able to use the mail system to send mail notifications to mail recipients and all of it is set up automatically. However, this is only part of the equation. Whenever you have an application sending email the other half of the equation is the receiving mail server. Often email notifications do not work so here are several solutions.

Solutions for the Nagios Server

1. Create a Fully Qualified Domain Name (FQDN) for the Nagios server

The FQDN looks like this in two parts: mail.example.com. The hostname is mail and the domain is represented in example.com, which when you combine the two becomes the FQDN. The mail server that receives the email from Nagios probably requires a FQDN. So your hostname on the Nagios server must satisfy this requirement.

On a CentOS system you can create a FQDN by editing two files. The first file to edit is /etc/sysconfig/network. Note

the “HOSTNAME” is specifically listed.

```
NETWORKING=yes
NETWORKING_IPV6=no
HOSTNAME=nagios.example.com
GATEWAY=192.168.3.1
```

The second file to edit is /etc/hosts. Note that the 127.0.0.1 represents the localhost. There are also two examples of the hostname “nagios.example.com” and “nagios”. These both must be listed. Of course change the name of the server and the domain to fit your situation.

```
127.0.0.1      nagios.example.com nagios localhost.localdomain localhost
::1            localhost6.localdomain6 localhost6
```

2. Check tcp_wrappers

One of the major reasons that mail may not be sent when you have a Nagios box set up is that tcp_wrappers could be stopping the ability of the localhost to send email. Edit the /etc/hosts.allow so that it has this line:

```
ALL: 127.0.0.1
```

That will allow the localhost to send mail to the Nagios administrators.

3. Allow Nagios to Resolve Using DNS

One of the tests that most mail servers perform is to see if the DNS resolves correctly for the sending mail server, in this case Nagios. If you create a DNS entry for Nagios so that your Nagios server resolves on the Internet, the mail sent from Nagios is more likely to be received. However, as is often the case, organizations may be more concerned about the security of Nagios so allowing DNS resolution may not be the best choice. In that case, relay the mail sent from Nagios to the corporate mail server.

4. Allow Nagios to Relay Mail Through a Mail Server

Often mail will not be delivered because the mail server that Nagios is sending to will not relay the mail sent from Nagios. By default all mail servers are designed to stop relays. Therefore the organization will need to configure the corporate mail server to be able to relay mail from the Nagios server.

Without making Nagios a full blown mail server, those four options should get the mail working.

Testing Your Mail

If you wanted to check to see if you can send mail use this procedure. First, send email from the Nagios server to a gmail account. Gmail is more likely to accept mail from a Nagios as it is not as restricted as the corporate mail server, probably. As a user run the mail command:

```
mail user@some_gmail_account
```

Here is what it will look like when you send mail from the command line(note you end the mail body with a “.” on a line by itself:

```
mail user@gmail.com
Subject: Testing Nagios Mail
```

```
This is a test of the Nagios mail notifications.
```

```
.
```

```
Cc:
```

That will send mail using Sendmail(CentOS) to the email address you specified. You can check your logs and you should see something like this indicating the mail was sent.

```
tail /var/log/maillog
Mar 27 06:52:45 nagios sendmail[24474]: n2RCqjn3024474: to=user@some_email.com,
ctladdr=root (0/0), delay=00:00:00, xdelay=00:00:00, mailer=relay, pri=30045,
relay=[127.0.0.1] [127.0.0.1], dsn=2.0.0, stat=Sent (n2RCqjDf024475 Message
accepted for delivery)
```

You can test mail locally on the Nagios server with telnet. Be sure to indicate you are connecting on port 25 and on the localhost as it will only allow connections on the localhost. The commands you need for telnet are highlighted. Be sure to use the domain of your Nagios server. The body of the email is the “DATA” and you end that with a “.” on a line by itself.

```
telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.localdomain (127.0.0.1).
Escape character is '^].
220 nag2.local.net ESMTP Sendmail 8.13.8/8.13.8; Sat, 13 Nov 2010 07:30:29 -0700
MAIL FROM:<test@example.com>
250 2.1.0 <test@example.com>... Sender ok
RCPT TO:<user@local.net>
250 2.1.5 <user@local.net>... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Testing the Mail System.
```

```
.
```

```
250 2.0.0 oADEUTI6002388 Message accepted for delivery
```

```
QUIT
```

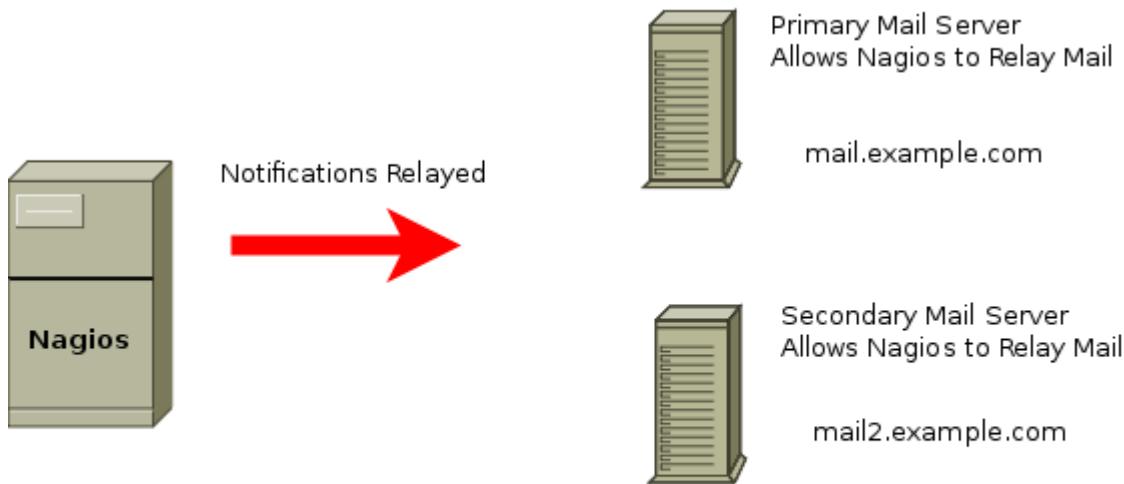
```
221 2.0.0 nag2.local.net closing connection
```

```
Connection closed by foreign host.
```

If you cannot telnet to the localhost on port 25 the mail server may not be working.

Redundant Mail Servers

When Nagios sends mail out it is important that Nagios relay through a mail server, or multiple mail servers, that will provide dependable delivery. The standard way to provide this is to have a primary and secondary mail server to insure the mail is relayed to the administrators. The illustration demonstrates this principle.



Redundant Mail Servers for Notifications

The mail server that is primary and the one that is secondary is determined by the DNS MX records for the domain.

	IN NS	ns.example.com.
example.com.	IN A	192.168.3.1
mail.example.com.	IN A	192.168.3.2
mail2.example.com.	IN A	192.168.3.3
example.com.	IN MX	10 mail.example.com.
example.com.	IN MX	20 mail2.example.com.

In this example mail.example.com is the primary because it has the lowest number. If the primary is not available the secondary will accept mail delivery.

Commit Error from the Web Interface

If you run into this error it is related to a permissions problem.

```
Error: Could not open command file '/usr/local/nagios/var/rw/nagios.cmd' for update!
The permissions on the external command file and/or directory may be incorrect.
Read the FAQs on how to setup proper permissions.
```

An error occurred while attempting to commit your command for processing.

Here are the permissions on named pipe, which are correct.

```
ls -l /usr/local/nagios/var/rw
total 0
prw-rw---- 1 nagios nagcmd 0 Mar  6 15:25 nagios.cmd
```

However, the problem relates to the apache process being able to make the commit change which would require the user apache to have write access to the pipe. To make that happen add apache to the nagcmd group in /etc/group.

```
nagcmd:x:501:nagios,apache
```

Now restart apache and you will see that it works fine.

Chapter 6: Monitoring

Nagios provides numerous methods to monitor a device whether that be with plugins or scripts which access clients using public ports, NRPE, SSH NSClient++, SNMP or even accepting passive checks from clients. Monitoring choices are based on network protocol requirements or administrative skills in configuring the options for monitoring.

This chapter provides a snapshot of some of those options for monitoring so that the student has basic framework for how these are set up. This information is not intended for a comprehensive approach to using each option.

Plugin Use

Plugins are used to gather information and return that information to the Nagios server. Each plugin will evaluate the situation and return a status value to Nagios. There are four status values that Nagios interprets.

0	OK	the stats is as expected
1	WARNING	a warning limit has been reached
2	CRITICAL	a critical limit has been reached
3	UNKNOWN	the status is unknown, misconfiguration

In order for Nagios to provide these four levels of status settings, warning and critical limits must be established. An important aspect of setting these limits is that each network will have different equipment and varying needs so these settings should reflect the individual network. Here are typical options for most plugins.

Typical Options

-h,	--help	Print detailed help screen
-V,	--version	Print version information
-H	--hostname=ADDRESS	Host name, IP Address, or unix socket (must be an absolute path)
-w	--warning=DOUBLE	Response time to result in warning status (seconds)
-c	--critical=DOUBLE	Response time to result in critical status (seconds)
-t	--timeout=INTEGER	Seconds before connection times out (default: 10)
-v	--verbose	Show details for command-line debugging (Nagios may truncate output)
-4	--use-ipv4	Use IPv4 connection
-6	--use-ipv6	Use Ipv6 connection

Monitoring Public Ports

Public ports are simply ports that are available to anyone, so they provide an easy way to monitor a host or service. However, the extent that a host can be monitored with public ports is limited to what can be discerned from the outside. Monitoring public ports, whether they may be TCP or UDP, is a basic option for Nagios. Keep in mind that firewalls can provide limited access to these public ports.

check_ping

Ping is a standard method of checking to see if a network device is up and this use of ICMP is the typical method used to monitor a host with Nagios. If the host does not respond to the ICMP check it is assumed the host is down.

Uniq Options

-p --packets=INTEGER number of ICMP ECHO packets to send (Default: 5)

Here is a service definition with a warning level of 60 milliseconds or 5% packet loss and a critical level of 100 milliseconds or 10% loss. This demonstrates that the settings need to be specific to the device or the network as networks vary. The default is 5 packets in the ping.

```
define service{
    use generic-service
    host_name centos
    service_description Ping
    check_command check_ping!60.0,5%!100.0,10%
}
```

The command definition can include the settings for warning and critical level if you want to make them standard for all uses of ping on a network.

```
define command{
    command_name check-host-alive
    command_line $USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c
5000.0,100% -p 5
}
```

check_tcp

This plugin will provide the flexibility you need if you need to monitor a port just to verify that the port is available. Here is an example of portmap service checks.

```
define service{
    use generic-service
    host_name centos
    service_description Portmap
    check_command check_tcp! 111
}

define command{
    command_name check_tcp
    command_line $USER1$/check_tcp -H $HOSTADDRESS$ -p $ARG1$ $ARG2$
}
```

Note that a common problem with check_tcp is that often the “-p” is added to the service definition. This will create the error “Port must be a positive integer” if the command definition already has the “-p”.

If you have any problems run the command from the command line to experiment.

```
./check_tcp -H 192.168.5.1 -p 111
TCP OK - 0.000 second response time on port 111|
time=0.000386s;;;0.000000;10.000000
```

check_smtp

Mail server communication on port 25 which is SMTP, Simple Mail Transfer Protocol. In order to check to see if the mail server is able to communicate with other mail server check port 25.

Uniq Options

-p	--port=INTEGER	Port number (default: 25)
-e	--expect=STRING	String to expect in first line of server response (default: '220')
-C	--command=STRING	SMTP command (may be used repeatedly)
-R	--command=STRING	Expected response to command (may be used repeatedly)
-f	--from=STRING	FROM-address to include in MAIL command, required by Exchange 2000
-F	--fqdn=STRING	FQDN used for HELO
-D	--certificate=INTEGER	Minimum number of days a certificate has to be valid.
-S	--starttls	Use STARTTLS for the connection.
-A	--authtype=STRING	SMTP AUTH type to check (default none, only LOGIN supported)
-U	--authuser=STRING	SMTP AUTH username
-P	--authpass=STRING	SMTP AUTH password

Here are two examples of service checks, one default on port 25 and the other on port 587.

```
define service{
    use                         generic-service
    host_name                   centos
    service_description          SMTP
    check_command                check_smtp
}
define service{
    use                         generic-service
    host_name                   centos
    service_description          SMTP Secure
    check_command                check_smtp!-p 587
}
```

This is the default command definition which allows the argument to use the port number.

```
define command{
    command_name    check_smtp
    command_line    $USER1$/check_smtp -H $HOSTADDRESS$ $ARG1$
```

Check from the command line to verify your check works as expected.

```
./check_smtp -H 192.168.5.1
SMTP OK - 0.002 sec. response time|time=0.002099s;;;0.000000
```

Or if you are using port 587 you can test with this check which just adds a different port.

```
./check_smtp -H 192.168.5.1 -p 587
SMTP OK - 0.012 sec. response timetime=0.011947s;;;0.000000
```

check_pop, check_spop, check_imap, check_simap, check_mysql

-p	--port=INTEGER	Port number (default: none)
-E	--escape	Can use \n, \r, \t or \ in send or quit string. Must come before send or quit
option	Default: nothing added to send, \\r\\n added to end of quit	
-s	--send=STRING	String to send to the server
-e	--expect=STRING	String to expect in server response (may be repeated)
-A	--all	All expect strings need to occur in server response. Default is any
-q	--quit=STRING	String to send server to initiate a clean close of the connection
-r -	--refuse=ok warn crit	Accept TCP refusals with states ok, warn, crit (default: crit)
-M	--mismatch=ok warn crit	Accept expected string mismatches with states ok, warn, crit (default: warn)
-j	--jail	Hide output from TCP socket
-m	--maxbytes=INTEGER	Close connection once more than this number of bytes are received
-d	--delay=INTEGER	Seconds to wait between sending string and polling for response
-D	--certificate=INTEGER	Minimum number of days a certificate has to be valid.
-S	--ssl	Use SSL for the connection.

check_imap

IMAP, Internet Message Access Protocol, is the interface that accesses mail on the mail server and provides access to user. There are several plugins that are options.

Of course you need to set up the hosts for the mail servers that you want to monitor. The first service is a check on IMAP port 143 with a timeout of five seconds and it searches for a text string “OK” which is part of the response of the IMAP server to a connection request. The “-e” is the expect string.

```
define service{
    use generic-service
    host_name lts
    service_description IMAP4 Response Check
    check_command check_imap!-t 5 -e "OK"
}

define command{
```

```
command_name      check_imap
command_line      $USER1$/check_imap -H $HOSTADDRESS$ $ARG1$
}
```

Here you can see a response from a mail server in the logs, that you can scan for a text string you want to test for.

```
nagios: SERVICE ALERT: lts;IMAP4 Dovecot Check;OK;SOFT;2;IMAP OK - 0.002 second
response time on port 143 [* OK [CAPABILITY IMAP4rev1 LITERAL+ SASL-IR LOGIN-
REFERRALS ID ENABLE STARTTLS LOGINDISABLED] Dovecot ready.]
```

check_simap

The service IMAPS shows the port listing as port 993 while it searches for a more detailed text string in “Dovecot ready”. That search string specifically indicates that Dovecot, the MDA (Mail Delivery Agent), is ready to communicate. Searches are case sensitive and you will need to place specific text in these searches that relate to your MDA and the ports you are using. The expect string "Dovecot ready" is an example for a Linux Dovecot daemon so it is important to add the string expected with the Mail Delivery Agent you are using.

```
define service{
    use                  generic-service
    host_name           ubpost
    service_description IMAPS Check
    check_command       check_simap!-p 993 -t 5 -e "Dovecot ready"
}

define command{
    command_name        check_simap
    command_line        $USER1$/check_simap -H $HOSTADDRESS$ $ARG1$
}
```

check_ftp

The basic command will connect on port 21 expecting a 220 reply from the FTP server. In this example, the return information not only shows the “220” but also the version of the FTP daemon (vsFTPd 2.0.5).

```
./check_ftp 192.168.5.1
FTP OK - 0.003 second response time on port 21 [220 (vsFTPd 2.0.5)]|time=0.002800s;;0.000000;10.000000

define command{
    command_name        check_ftp
    command_line        $USER1$/check_ftp -H $HOSTADDRESS$ $ARG1$
}

define service{
```

```

use generic-service
host_name centos
service_description FTP
check_command check_ftp
}

}

```

Modifications

If you wanted to lower the timeout from the default 10 seconds to 4 seconds by using the “-t” timeout option and adding the timeout. By changed the expect string “-e” to “vsFTPD 2.0.5” Nagios will send notifications if the string does not show up meaning either the FTP server is down or the version has changed, both are valuable information.

```
check_command check_ftp! -t 4 -e "vsFTPD 2.0.5"
```

check_http

A common public port that often is checked is port 80, http. There are a significant number of options with this plugin to get out of it as much as possible.

-I --IP-address=ADDRESS	IP address or name (use numeric address if possible to bypass DNS lookup).
-p --port=INTEGER	Port number (default: 80)
-S --ssl	Connect via SSL. Port defaults to 443
--sni	Enable SSL/TLS hostname extension support (SNI)
-C --certificate=INTEGER	Minimum number of days a certificate has to be valid. Port defaults to 443
-e, --expect=STRING	Comma-delimited list of strings, at least one of them is expected in the first (status) line of the server response (default: HTTP/1.) If specified skips all other status line logic (ex: 3xx, 4xx, 5xx processing)
-s --string=STRING	String to expect in the content
-u --url=PATH	URL to GET or POST (default: /)
-P --post=STRING	URL encoded http POST data
-j --method=STRING	(HEAD, OPTIONS, TRACE, PUT, DELETE) Set HTTP method.
-N --no-body	Don't wait for document body: stop reading after headers.
-M --max-age=SECONDS	Warn if document is more than SECONDS old. the number can also be of the form "10m" for minutes, "10h" for hours, or "10d" for days.
-T --content-type=STRING	specify Content-Type header media type when POSTing
-l --linespan	Allow regex to span newlines (must precede -r or -R)
-r --regex, --ereg=STRING	Search page for regex STRING
-R --eregi=STRING	Search page for case-insensitive regex STRING
--invert-regex	Return CRITICAL if found, OK if not
-a --authorization=AUTH_PAIR	Username:password on sites with basic authentication
-b --proxy-authorization=AUTH_PAIR	Username:password on proxy-servers with basic authentication
-A --useragent=STRING	String to be sent in http header as "User Agent"
-k --header=STRING	Any other tags to be sent in http header. Use multiple times for additional headers
-L --link	Wrap output in HTML link (obsoleted by urlize)
-f --onredirect=<ok warning critical follow sticky stickyport>	
-m, --pagesize=INTEGER<:INTEGER>	Minimum page size required (bytes) : Maximum page size required (bytes)

This is the standard way to use the check_http. It checks to verify communication is available on port 80 of a web server. This is in fact, a better check on the server than the check_ping which can only determine if the server is up. This simple check provides some peace of mind and a place to start.

```
define service{
    use                         generic-service
    host_name                   centos
    service_description          HTTP
    check_command                check_http
}
}
```

These two checks are related to the SSL options with the web server. Note that the checks change to port 443 if you use the --ssl option, they are testing to see if the web server can serve secure pages and if the web server certificate is valid for the next 21 days. The first check will test for a response within a limited time frame, 5 seconds for a warning or more than 10 seconds for a critical state.

```
define service{
    use                         generic-service
    host_name                   centos
    service_description          Secure HTTP
    check_command                check_http! -w 5 -c 10 --ssl
}
}
```

This check is focused on the certificate. In this example, if the certificate is good for more than 21 days an “OK” is returned. A WARNING state is triggered if the certificate has less than 21 days before it expires. A CRITICAL state is triggered when the certificate has expired.

```
define service{
    use                         generic-service
    host_name                   centos
    service_description          Certificate
    check_command                check_http! -C 21
}
}
```

Both of the service checks above will return the following output in the Nagios web interface.
OK - Certificate will expire on 05/25/2012 23:59.

This usage of check_http allows you to check to see if a directory requiring authorization with username and password is working correctly. Note that the check_http has been redefined to check_http_auth so that additional arguments can be used. The service definition includes the IP Address of the server, the directory that requires authentication (-u/sales) and the username and password required to access the directory. Each is separated by a “!”. Note the command definition included.

```
define service{
    use                         generic-service
    host_name                   centos
    service_description          Sales Authorization
}
}
```

```

check_command          check_http_auth!192.168.5.1 -u/sales!tom!
user_password
}

define command{
    command_name      check_http_auth
    command_line      $USER1$/check_http -H $ARG1$ -a $ARG2$:$ARG3$
}

```

If the user login is not correct warning will be issued with the “401 Authorization Required”. This enables you to verify password changes and integrity. However, leaving a plain text password in the Nagios config files is not the best idea.

The check_http plugin can also be used to monitor content on a web page.

check_mysql

If you had a Wordpress blog with a MySQL database you could monitor that database from a public port or you can monitor internally and return the information to Nagios using NRPE.



Security Tip

The first thing you want to make sure of is that you have placed a password on the root account for MySQL.

```
mysqladmin -h localhost -u root password "your_password"
```

Note: If you compile the Nagios plugins, you must install packages for mysql if you want to use the check_mysql plugin.

```
yum install -y mysql mysql-devel
```

Host to be Monitored

If your Wordpress database was called "word" you could allow read rights to the nagios user from the IP Address of your Nagios server.

```
GRANT select ON word.* TO nagios@192.168.5.51;
```

This setting is important to only allow “select” on the
Edit the commands.cfg file to create a command option that represents your database.

```

define command{
    command_name      check_mysql
    command_line      $USER1$/check_mysql -H $HOSTADDRESS$ -u nagios -d word
}

```

Edit the services.cfg file to create a service definition.

```
define service{
    use                      generic-service
    host_name                centos
    service_description        Wordpress_Database
    check_command              check_mysql
}
}
```

Another way to design this is to allow the administrator to enter different users and databases if the need was to monitor several different databases.

```
define command{
command_name    check_mysql
command_line     $USER1$/check_mysql -H $HOSTADDRESS$ -u $ARG1$ -d $ARG2$
}

define service{
    use                      generic-service
    host_name                centos
    service_description        Wordpress_Database
    check_command              check_mysql!nagios!word
}
}
```

Checking multiple databases on the same server will require the same basic steps as before with MySQL and you will probably want a different user. Here the database name is db7 and the user is webadmin.

```
GRANT select ON db7.* TO webadmin@192.168.5.51;
```

Here you can use the same command definition but you will need to change the service_description and the check_command to fit the database you want to check.

```
define service{
    use                      generic-service
    host_name                centos
    service_description        DB7_Database
    check_command              check_mysql!webadmin!db7
}
}
```

The command “show status;” once you have selected a database provides similar output as the information provided when Nagios collects a summary.

```
show status;
+-----+-----+
| Variable_name          | Value   |
+-----+-----+
---cut---
| Bytes_received          | 428    |
+-----+-----+
```

Bytes_sent	14514
--cut--	
Com_show_databases	1
Com_show_errors	0
Com_show_fields	11
--cut--	
Com_show_processlist	1
--cut--	
Compression	OFF
Connections	62
--cut--	
Handler_read_rnd_next	267
Handler_rollback	0
Handler_savepoint	0
Handler_savepoint_rollback	0
Handler_update	0
Handler_write	396
Innodb_buffer_pool_pages_data	19
Innodb_buffer_pool_pages_dirty	0
Innodb_buffer_pool_pages_flushed	0
Innodb_buffer_pool_pages_free	493
Innodb_buffer_pool_pages_misc	0
Innodb_buffer_pool_pages_total	512
Innodb_buffer_pool_read_ahead_rnd	1
Innodb_buffer_pool_read_ahead_seq	0
Innodb_buffer_pool_read_requests	77
Innodb_buffer_pool_reads	12
--cut--	
Innodb_data_read	2494464
Innodb_data_reads	25
Innodb_data_writes	3
Innodb_data_written	1536
Innodb dblwr_pages_written	0
Innodb dblwr_writes	0
Innodb_log_waits	0
Innodb_log_write_requests	0
Innodb_log_writes	1
Innodb_os_log_fsyncs	3
Innodb_os_log_pending_fsyncs	0
Innodb_os_log_pending_writes	0
Innodb_os_log_written	512
Innodb_page_size	16384
Innodb_pages_created	0
Innodb_pages_read	19
--cut--	
Key_blocks_unused	7189
Key_blocks_used	56
Key_read_requests	3128
Key_reads	56
Key_write_requests	494
Key_writes	307
Last_query_cost	10.499000

```

| Max_used_connections | 5 |
| Not_flushed_delayed_rows | 0 |
| Open_files | 34 |
| Open_streams | 0 |
| Open_tables | 17 |
|---cut---|
| Queries | 1439 |
| Questions | 19 |
| Rpl_status | NULL |
| Select_full_join | 0 |
| Select_full_range_join | 0 |
| Select_range | 0 |
| Select_range_check | 0 |
| Select_scan | 4 |
| Slave_open_temp_tables | 0 |
| Slave_retried_transactions | 0 |
| Slave_running | OFF |
| Slow_launch_threads | 0 |
|---cut---|
| Threads_created | 61 |
| Threads_running | 1 |
| Uptime | 19367 |
| Uptime_since_flush_status | 1439 |
+-----+
249 rows in set (0.00 sec)

```

Testing for Page Content

Nagios can be used to actually verify that content can be read on the web page. So not only will you know that the web server is up and the port open but you can verify that a specific page is readable.

```

/usr/local/nagios/libexec./check_http -H 192.168.5.1 -u /2011/01/09/nagios-
test/ -s "Nagios Test"
HTTP OK: HTTP/1.1 200 OK - 9381 bytes in 0.344 second response time |
time=0.344154s;;;0.000000 size=9381B;;;0

```

The check_http plugin is used followed by the host, in the command definition. The “-u /2011/01/09/nagios-test/” is the page. Since this is a Wordpress site it contains the folder that you see before the page itself. Note that the page is followed by a “/”.

```

define command{
    command_name      check_http_page
    command_line      $USER1$/check_http -H $HOSTADDRESS$ -u /2011/01/09/nagios-
test/ -s "Nagios Test"
}

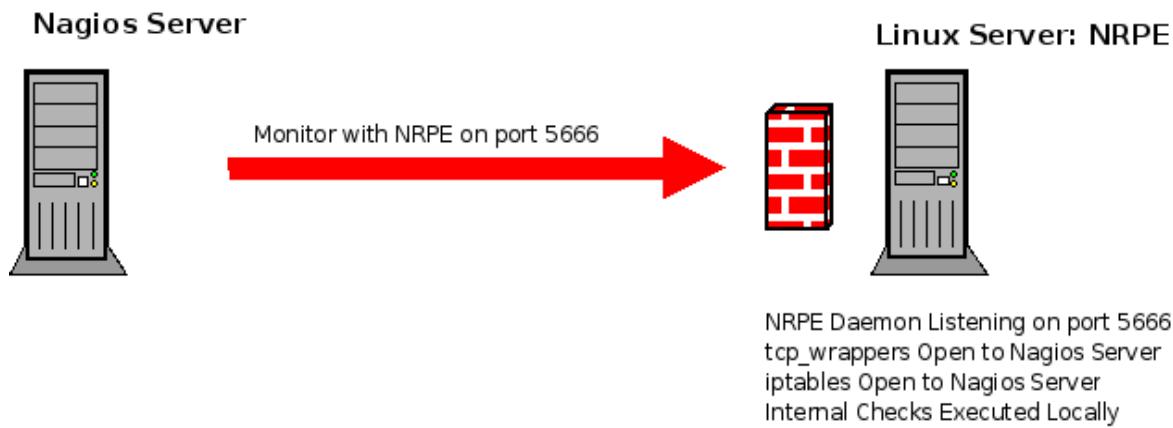
define service{
    use                  generic-service
    host_name           centos
    service_description Wordpress_Page
    check_command       check_http_page
}

```

Monitoring Linux

The Nagios Remote Plugin Executor or NRPE allows you to execute programs for monitoring purposes on the remote server. One advantage of NRPE is that it does not require a login to perform the tests on the remote server. NRPE allows you to monitor internal aspects of a Linux server from the Nagios server. When you monitor public ports like HTTP you can determine if the web server is running by using these service checks, but you are not able to monitor other aspects of the server which you may need information on, which is why you will want to use NRPE.

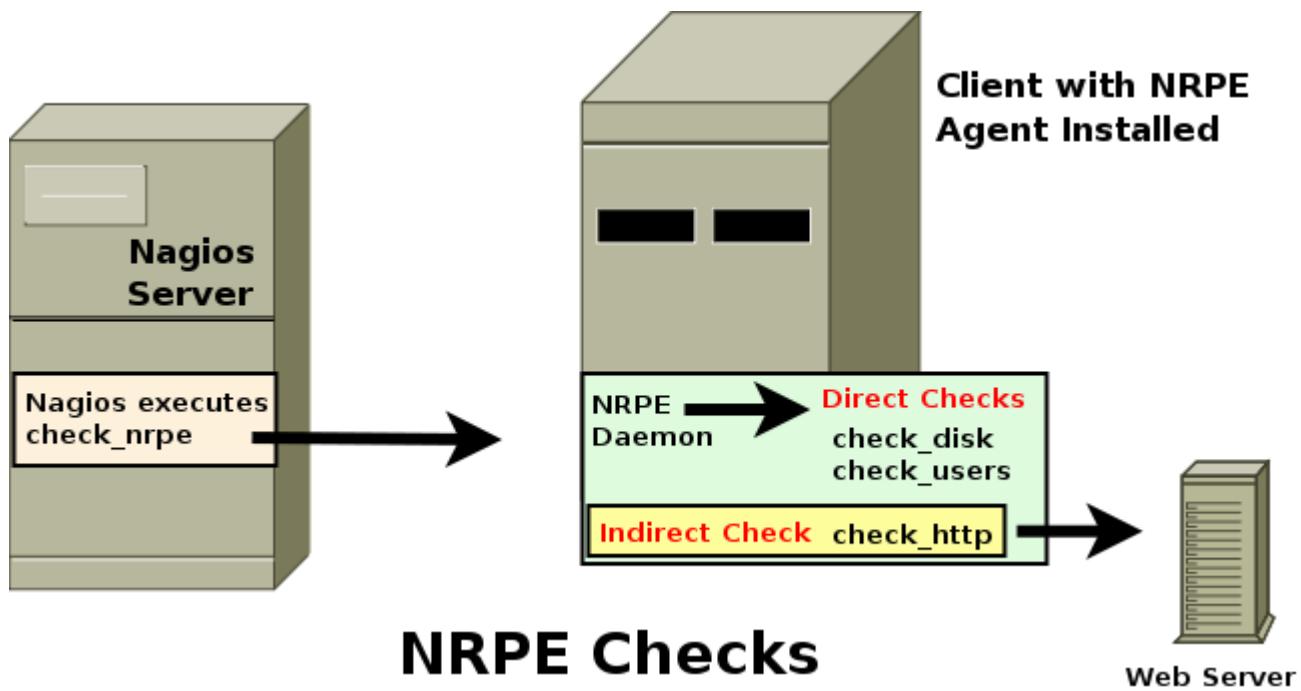
Monitoring With NRPE



NRPE Concepts

NRPE is able to perform two types of checks, Direct and Indirect. In direct NRPE checks the Nagios server executes check_nrpe which then connects to the NRPE daemon which is running on the client. The NRPE daemon then will execute the command that was requested from the Nagios server. This command could execute a plugin locally as is illustrated in the example. The command could also execute any script on the client whether it be a bash shell script, a perl script or any other type of script.

The example also illustrates that NRPE can be used to execute Indirect Checks. This may be a situation where the Nagios server has access to the client but not the web server that needs to be monitored. However, the client may have access to the web server. So in this example the Nagios server executes a plugin or script on the client which in turn monitors another box on the network, thus Nagios is indirectly monitoring the web server.



Install the Daemon xinetd

A daemon, or service, is a background process that performs a specific function. Originally all daemons started at boot time, running in the background and providing services when called upon. However, as the number of daemons grew, the resources that these daemons took up increased and began slowing down the server so developers began looking for a way to limit the number of daemons that run on a system. The daemon inetd was developed to help manage other daemons by listening for requests in their behalf. When a request was received for a daemon managed by inetd it was then notified and started up to respond. This has saved considerable resources on servers and it has increased the interest in superdaemons and methods of reducing resources.

The xinetd superdaemon has replaced inetd on most Linux distributions today. xinetd has become more popular because of security restrictions that can be placed on those who access the daemons managed by xinetd. xinetd also provides better protection from denial of service attacks, better log management, and more flexibility. Both inetd and xinetd only work with daemons that provide connections over a network.

How to Protect the NRPE Daemon

Server daemons must be protected to be effective. There is no perfect or complete option, but there are definite ways to minimize the risk.

1. Limit Connections to Daemons

Connections to daemons can be limited by using several powerful tools. Iptables firewall is probably the most flexible and powerful tool than an administrator has access to. However, it is at the same time the most complex. Tcp_wrappers is a tool that is easy to use and works with most daemons to limit access to daemons to specific subnets or IP Addresses.

2. Limit the Number of Connections

Your network and hardware can only handle a limited number of connections safely. When connections push your resources to the limit you will often see vulnerabilities appear that would not normally exist. When resources begin to fail some options and security programs cannot function to their full extent.

You will need to install xinetd and make sure you have a file in /etc/xinetd.d called nrpe on the client and it looks like this:

```
# default: off
# description: NRPE (Nagios Remote Plugin Executor)
service nrpe
{
    flags          = REUSE
    type           = UNLISTED
    port           = 5666
    socket_type   = stream
    wait           = no
    user           = nagios
    group          = nagios
    server         = /usr/sbin/nrpe
    server_args    = -c /usr/local/nagios/etc/nrpe.cfg --inetd
    log_on_failure += USERID
    disable        = no
    only_from      = 127.0.0.1 192.168.5.50
}
```

These are the two most important lines. By default all daemons monitored by xinetd are disabled so the default line says “disable = yes”. The “only_from” line allows you to determine which machines can monitor this server using NRPE, this is where you will enter the IP Address for the Nagios server as well as the localhost.

```
    disable        = no
    only_from      = 127.0.0.1 192.168.5.50
```

Edit /etc/services and add this line:

```
nrpe           5666/tcp          # Nagios Remote Monitoring
```

Restart xinetd and view the log at /var/log/daemon.log

```
service xinetd restart
tail /var/log/daemon.log
```

Look for errors to correct.

Edit the /usr/local/nagios/etc/nrpe.cfg (RPM repository /etc/nagios/nrpe.cfg).

Change your allowed_hosts address to reflect the nagios monitoring server. You should also allow the localhost so that you can do testing if necessary. If you are using xinetd then this does not have any impact.

```
allowed_hosts=127.0.0.1 192.168.5.180
```

The basic plugins that are running for you initially are these listed below. Note the path will be different if you installed using the RPM repository.

```
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
command[check_load]=/usr/local/nagios/libexec/check_load -w 15,10,5 -c 30,25,20
command[check_hda1]=/usr/local/nagios/libexec/check_disk -w 20 -c 10 -p /dev/hda1
command[check_zombie_procs]=/usr/local/nagios/libexec/check_procs -w 5 -c 10 -s Z
command[check_total_procs]=/usr/local/nagios/libexec/check_procs -w 150 -c 200
```

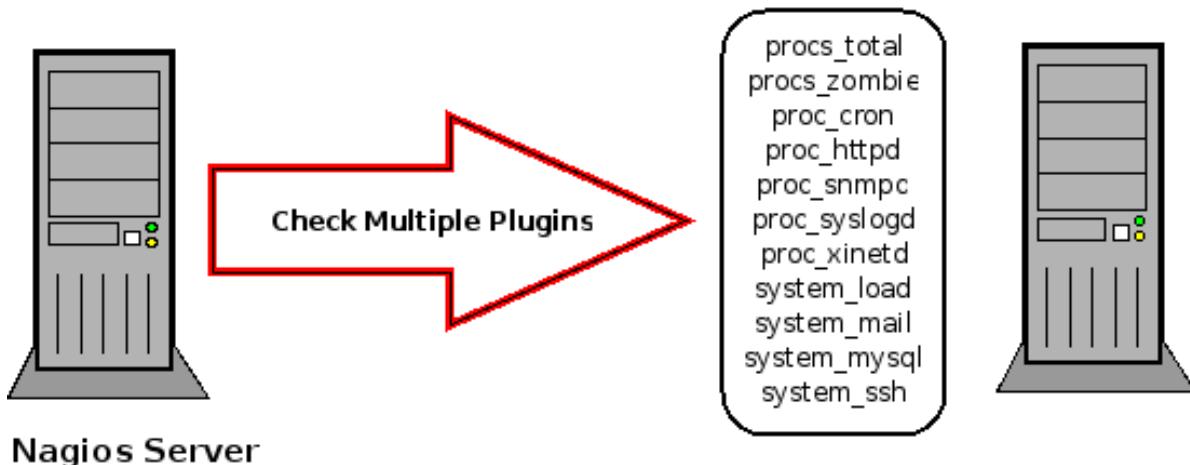
Change ownership on the /usr/local/nagios/etc/nrpe.cfg (RPM repository /etc/nagios/nrpe.cfg) file so that nagios is able to read the file.

```
chown nagios /usr/local/nagios/etc/nrpe.cfg
```

SSH Concepts

Generally there are three good reasons for using SSH to monitor remote Linux servers. First, of course the **data transmission and password authentication is always encrypted**. This makes it an excellent choice when you are located in a hostile environment. The second reason for using SSH is that you can **employ the check_multi command** which allows you to check almost an unlimited number of services with one check. This is a powerful way to save on bandwidth as you are only making one check for multiple services and the process is encrypted.

Scale Nagios with Multiple Checks at One Time



The final reason you may consider SSH to the checking on the remote server is that you may not be able to install applications or compile applications on the remote server.

In order to use SSH you need to use the check_by_ssh plugin which will enable you to perform local checks on the remote machine.

Configure the Nagios Server

There are a number of steps that you must take to prepare the Nagios server to be able to use SSH without using a password each time. This involves setting up a SSH keypair for the nagios user and providing the public key for the nagios user on the remote boxes that will be monitored using SSH.

Using SSH to Check Services

Now you can set up individual checks using SSH. Here are several services listed in the services.cfg. Note several differences with other services that you already have. The check_command is different showing that it is a SSH command that will run the normal check. What this means is that you must edit the commands.cfg and create each one of these SSH commands that you will use.

```
define service{
    use          generic-service
    host_name   class
    service_description  SSH Check Disk
    check_command   check_ssh_disk!10%!5%! /dev/mapper/VolGroup00-
LogVol00
}
```

Here is the listing in the commands.cfg. It illustrates how you must configure the command to use the SSH key so that you do not need to use a password and it functions just like the check_disk but you must configure it to use SSH. Note that this check is testing for the disk usage on this partition which is a logical volume. Make sure you understand the description of your partition.

```
dev/mapper/VolGroup00-LogVol00

define command {
    command_name      check_ssh_disk
    command_line      $USER1$/check_by_ssh -H $HOSTADDRESS$ -i
/usr/local/nagios/etc/.ssh/id_dsa -C "$USER1$/check_disk -w $ARG1$ -c $ARG2$ -p
$ARG3$"
}
```

This service listing in services.cfg checks for system load.

```
define service{
    use          generic-service
    host_name   class
    service_description  SSH Check Load
    check_command   check_ssh_load!5.0,4.0,3.0!10.0,6.0,4.0
}
```

Here is the corresponding command that must be placed in commands.cfg.

```
define command {
    command_name      check_ssh_load
    command_line      $USER1$/check_by_ssh -H $HOSTADDRESS$ -i
/usr/local/nagios/etc/.ssh/id_dsa -C "$USER1$/check_load -w $ARG1$ -c $ARG2$"
}
```

Monitoring Windows

The Windows client NSClient++ can be used monitor a Windows machine with NSClient++ using the check_nt command or using NRPE. Because the configuration for both aspects involves the NSClient++ they are viewed together. The first step in setting up NRPE for Windows is to download a client for the Windows machine.

Download the NSClient++ from <http://sourceforge.net/projects/nscplus>

This will provide a .zip file which you can unzip and it will provide the NSClient++-Win32-x.x.x folder.

NSClient++ Concepts

Login as the Administrator to the server.

Create a directory under the [C:\](#) drive and download NSClient++ into that drive. Unzip the file and enter the directory that was created.

Once you install you will have to make a note of the location of the install directory path.

Here is the contents of the directory.

Name	Date modified	Type	Size	Tags
modules	10/4/2010 11:44...	File Folder		
scripts	10/4/2010 11:44...	File Folder		
changelog	5/27/2010 10:50...	Text Document	56 KB	
counters.defs	5/27/2010 9:30 PM	DEFS File	9 KB	
license	5/27/2010 9:30 PM	Text Document	18 KB	
Nagios Usage Guide....	5/27/2010 9:30 PM	PDF File	901 KB	
nsc	5/27/2010 9:30 PM	Configuration Se...	12 KB	
NSClient++	5/27/2010 10:50...	Application	420 KB	
NSClient++.pdb	5/27/2010 10:50...	PDB File	3,396 KB	
NSClient++ Referen...	5/27/2010 9:30 PM	PDF File	872 KB	
+nstray	5/27/2010 10:50...	Application	272 KB	
nstray.pdb	5/27/2010 10:50...	PDB File	2,155 KB	
readme	5/27/2010 9:30 PM	Text Document	2 KB	

On the Windows machine place the path for the .exe file in the run command and install the program.

```
C:\NSClient++-Win32-0.3.8\NSClient++.exe /install
```

You will see a security warning but continue the install.

Now start the program, note your path may be different.

```
C:\NSClient++-Win32-0.3.8\NSClient++.exe /start
```

In order to stop the program use this command.

```
C:\NSClient++-Win32-0.3.8\NSClient++.exe /stop
```

You can test with:

```
C:\NSClient++-Win32-0.3.8\NSClient++.exe /test
```

If you make any changes to the configuration, stop the service and restart it.

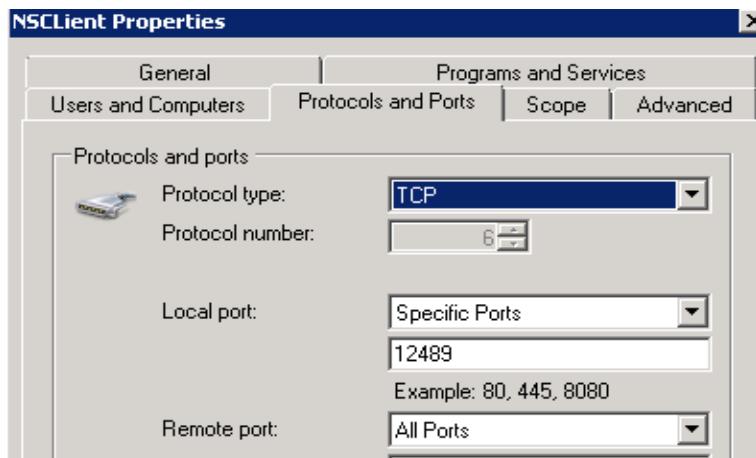
Edit the NSC.ini file that is in the NSClient directory. Note that the file is divided by keywords placed in brackets. First go to the [modules] section and edit the checks that you want to use. Uncomment the lines that you see below. The FileLogger.dll will log the activities of the NSClient++. CheckDisk.dll will check for file size and hard disk use. The CheckSystem.dll will check for memory, uptime, service stats and processes. You will also need to uncomment the NSClientListener.dll and the NRPEListener.dll in order to communicate with Nagios.

In order to use some of the options available with NSClient++ you need to allow additional features. There are characters that need to be used with commands `&<>` `[]{}{}` that you will want to allow, “nasty_meta_chars”. The “allow_arguments” will allow NRPE parameters to be passed along. Now there are some security issues with enabling this option so you need to consider that factor.

Go to the global section, [Settings], and be sure to limit the access to the Windows server that you are going to monitor. Under the Allowed Hosts section enter the local host and any other connections that you want to enable. These addresses will be separated by a comma.

```
allowed_hosts=127.0.0.1/32,192.168.5.50
```

In the Windows firewall open two ports, 5666 for NRPE and 12489 for NSClient++. Both are TCP ports. You can see in the example how it should look when you review the firewall.



Limit access to these ports to the Nagios server only.

Option	Description
port=5666	port the NRPEListener.dll will listen to
command_timeout=60	maximum number of seconds that the NRPE daemon will allow plug-ins to complete check
allow_arguments=1	allow clients to specify arguments to commands that are executed
allow_nasty_meta_chars=1	allow clients to specify nasty (`&><"\[\{\}]") characters
use_ssl=0	boolean option to use a SSL connection
bind_to_address=	bind to specific address, blank means all addresses
allowed_hosts=192.168.1.1	Security option to limit access to Windows server and NRPE
script_dir=scripts\	Files in this directory become check commands, has security implications
socket_timeout=30	Socket time to close incoming connections

The NRPE Handlers represent the actual commands that will be used.



Unfortunately this section is deprecated and it is not recommended to use these settings.

```
[NRPE Client Handlers]
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
check_disk1=/usr/local/nagios/libexec/check_disk -w 5 -c 10
check_disk_c=inject CheckFileSize ShowAll MaxWarn=1024M MaxCrit=4096M
```

Once you have the remote host set up you will need to set up the Nagios monitoring server. First install the nrpe plugin.

MSSQL

Monitoring a Windows SQL server is an aspect that many Nagios administrators will need to do. One of the better plugins for this is check_mssql_health. This plugin provides health related checks that are useful in understanding the status of a SQL server.

Service Definitions

In this example the host definition is listed with service definitions which may be the case if you are only checking one server. These all use the port 1433 so one check is designed to monitor the port itself with a tcp check.

```
define host{
    use                  windows-server
    host_name           win2008
    alias               Windows Server
    address             184.106.77.67
}

define service{
    use                  generic-service
    host_name           win2008
    service_description SQL Connectivity 1433
    check_command       check_tcp!1433
}

define service{
    use                  generic-service
    host_name           win2008
    service_description SQL io-busy
    check_command       check_mssql _health!io-busy
}

define service{
    use                  generic-service
    host_name           win2008
    service_description SQL Backup of model
    check_command       check_mssql _health!database-backup-age
--name model
}

define service{
    use                  generic-service
    host_name           win2008
    service_description SQL Page Life
    check_command       check_mssql _health!page-life-expectancy
}

define service{
    use                  generic-service
}
```

```

host_name          win2008
service_description SQL Locks
check_command      check_mssql _health!list-locks
}

define service{
    use           generic-service
    host_name    win2008
    service_description SQL Datafiles
    check_command   check_mssql _health!list-datafiles
}

define service{
    use           generic-service
    host_name    win2008
    service_description SQL Databases
    check_command   check_mssql _health!list-databases
}

define service{
    use           generic-service
    host_name    win2008
    service_description SQL Connected Users
    check_command   check_mssql _health!connected-users
}

```

It is best practice to test from the command line first. Be sure the Windows server firewall is open on port 1433 for the Nagios server only as the username and password are plain text.

Log Monitoring

The basic plugin for analyzing logs is check_log. The check_log will check a current log for a text string and after the check save the information in a temporary log so that it can compare changes. This plugin has two settings; 0 for “OK” and 2 for “Critical”. Here is an example check.

```
check_log -F /var/log/messages -O /tmp/check_log.fail -q "Warning"
```

The (-F) will point to the log you want to monitor. The (-O) refers to the “old log” allowing the system to compare. The (-q “text_string”) is what you will be searching for in the log.

Here is a return from that command which shows that it found 2 instances:

```
/usr/local/nagios/libexec./check_log -F /var/log/messages -O /tmp/check_log.fail
-q "Warning"
(2) < Apr 28 14:28:03 fw3 nagios: Warning: Return code of -1 for check of service
'Traffic Load OUT' on host 'cisco2500' was out of bounds.
```

Monitor Nagios Logs

In this example the commands that are defined will be used for a local check of the Nagios logs. Each one is defined separately as you will actually need different commands for each text string you are looking for.

Edit /etc/nagios/commands.cfg

```
define command{
    command_name      check_log_warning
    command_line $USER1$/check_log -F /var/log/messages -O /tmp/check_log.fail
-q "Warning"
}
define command{
    command_name      check_log_fail
    command_line $USER1$/check_log -F /var/log/messages -O /tmp/check_log.fail
-q "Fail"
}
```

Service Definition

Create a service definition for each command you will use.

```
define service{
    use                      local-service
    host_name                localhost
    service_description       Check Log
    check_command             check_log_fail
}
```

Nagios Server Logs

You should see examples in the logs that demonstrate the check is working. You can see the process of a check as it is exhibited step by step in the log.

```
Apr 26 15:01:16 fw3 xinetd[2068]: START: nsca pid=4334 from=192.168.5.91
Apr 26 15:01:16 fw3 xinetd[2068]: EXIT: nsca status=0 pid=4334 duration=0(sec)
Apr 26 15:01:16 fw3 nagios: EXTERNAL COMMAND:
PROCESS_SERVICE_CHECK_RESULT;passnag;Log;0;Log check data initialized...
Apr 26 15:01:17 fw3 nagios: PASSIVE SERVICE CHECK: passnag;Log;0;Log check data initialized...
Apr 26 15:02:12 fw3 xinetd[2068]: START: nsca pid=4418 from=192.168.5.91
Apr 26 15:02:12 fw3 xinetd[2068]: EXIT: nsca status=0 pid=4418 duration=0(sec)
Apr 26 15:02:12 fw3 nagios: EXTERNAL COMMAND:
PROCESS_SERVICE_CHECK_RESULT;passnag;Log;0;Log check ok - 0 pattern matches found
```

Network Printers

Nagios allows you to monitor network printers so that you will easily be able to verify basic information about your network printers. This will allow you to check if the printer is up and basic toner states. Why use Nagios for printers if you can log into a network printer? It provides one central location for checking on servers, printers, routers, etc. And it will notify according to your settings.

Edit the /usr/local/nagios/etc/nagios.cfg (RPM repository /etc/nagios/nagios.cfg) and uncomment the printer line.

```
#cfg_file=/usr/local/nagios/etc/objects/printer.cfg
```

Create the file for printers

Modify the template and enter the information for the printer that you want to use. The information here is for an HP 4300dn printer.

```
#####
#
# HOST DEFINITIONS
#
#####

define host{
    name          generic-printer
    use           generic-host
    check_period  24x7
    check_interval 5
    retry_interval 1
    max_check_attempts 10
    check_command  check-host-alive
    notification_period workhours
    notification_interval 30
    notification_options d,r
    contact_groups admins
    register      0          ; JUST A TEMPLATE
}

# Define a host for the printer we'll be monitoring
# Change the host_name, alias, and address to fit your situation

define host{
    use          generic-printer
    host_name    hp4300
    alias        HP LaserJet 4300dn
    address     192.168.5.11
    hostgroups   network-printers
}

#####

```

```

#
# HOST GROUP DEFINITIONS
#
#####
# A hostgroup for network printers

define hostgroup{
    hostgroup_name  network-printers           ; The name of the hostgroup
    alias          Network Printers            ; Long name of the group
}

#####
#
# SERVICE DEFINITIONS
#
#####

# Create a service for monitoring the status of the printer
# Change the host_name to match the name of the host you defined above
# If the printer has an SNMP community string other than "public", change the #
check_command directive to reflect that

define service{
    use           generic-service      ; from a template
    host_name     hp4300             ; host
    service_description Printer Status   ; service description
    check_command  check_hpjd!-C public ; command
    normal_check_interval 10          ; Check 10 minutes
    retry_check_interval 1           ; Re-check
}

#
# Create a service for "pinging" the printer occasionally. Useful for
# monitoring RTA, packet loss, etc.

define service{
    use           generic-service
    host_name     hp4300
    service_description PING
    check_command  check_ping!3000.0,80%!5000.0,100%
    normal_check_interval 10
    retry_check_interval 1
}

```

Once you have created the printer.cfg file be sure to restart nagios.

```
service nagios restart
If you get errors you will need to fix them.
```

The services to check for the printer are status and toner using the `check_hpjd`. This is a very common program that is easy to use. There are other printer checking programs that you can find at <http://www.nagiosexchange.org>.

Here you can see the Ping status and the printer status for the HP 4300.

Host ↑↓	Service ↑↓	Status ↑↓	Last Check ↑↓	Duration ↑↓	Attempt ↑↓	Status Information
dg	Current Users	OK	2009-01-16 15:49:34	0d 7h 15m 14s	1/4	USERS OK - 2 users currently logged in
gateway	PING	OK	2009-01-16 15:47:23	22d 23h 40m 30s	1/4	PING OK - Packet loss = 0%, RTA = 1.15 ms
hp4300	PING	OK	2009-01-16 15:44:55	0d 0h 15m 18s	1/4	PING OK - Packet loss = 0%, RTA = 6.18 ms
	Printer Status	OK	2009-01-16 15:49:44	0d 0h 12m 29s	1/4	Printer ok - ("Ready")

If you select the Printer Status in the web interface you will see additional information about the printer.

Checking Printers with SNMP

The `check_snmp_printer_args.sh` plugin is an excellent choice if you want generic options using SNMP (Simple Network Management Protocol). In most cases this will provide all of the information that you need.

Download the plugin SNMP Printer Check:

<http://exchange.nagios.org/directory/Plugins/Hardware/Printers/SNMP-Printer-Check/details>

Here is an example of the output that is provided clearly indicating toner level, page count, maintenance kit level and tray status.

Preparation

The script was written on a Windows machine so you will need to convert it. Install these two programs, bc is used by the script.

```
yum install -y dos2unix bc
```

Convert the program with:

```
dos2unix check_snmp_printer_args.sh
```

Make it executable and change ownership:

```
chmod 755 check_snmp_printer_args.sh
chown nagios:nagios check_snmp_printer_args.sh
```

Define the command in the commands.cfg

```
define command{
    command_name      check_snmp_printer_args
    command_line      $USER1$/check_snmp_printer_args.sh -H $HOSTADDRESS$ $ARG1$ $ARG2$
}
}
```

Create service definitions based on options that are available.

```
CONSUM {<string> | TEST | ALL}
TEST will give you the exact names of all available consumables
CONSUMX <string>
TEST will give you the #'s of all trays available
DISPLAY      Report contents of printer display
DEVICES       Status of hardware modules
MESSAGES      Event logs reported by the printer
MODEL         ALL will give you all tray output at once.
PAGECOUNT     How many pages this printer has processed (culmulative)
STATUS         Overall status of the printer
TRAY {<number> | TEST | ALL}      <number> will give you output for the
specified tray. A comma-separated list of values is possible as well.
```

Here you can see that two network printers are monitored each are HP, one is color and one is not. They both have been easily detected and the plugin provides excellent information.

hp4300	PING	OK	12-07-2011 05:32:52	0d 0h 37m 54s	1/3	PING OK - Packet loss = 0%, RTA = 6.03 ms
	Printed Pages	OK	12-07-2011 05:34:06	0d 0h 36m 40s	1/3	Pagecount is 1,206,990
	Printer Consumables	OK	12-07-2011 05:35:20	0d 0h 35m 26s	1/3	Black Print Cartridge HP Q1339A is at 56% - OK! Maintenance Kit HP 110V-Q2436A, 220V-CE2437A is at 87% - OK!
	Printer Model	OK	12-07-2011 05:36:34	0d 0h 34m 12s	1/3	hp LaserJet 4300, Serial # CNGY510590
	Printer Status	OK	12-07-2011 05:37:48	0d 0h 32m 58s	1/3	Printer ok - ("Powersave on")
	Printer Tray 2	OK	12-07-2011 05:39:02	0d 0h 31m 44s	1/3	TRAY 2 is OK!
hp_cp4525	PING	OK	12-07-2011 05:31:49	0d 0h 28m 57s	1/3	PING OK - Packet loss = 0%, RTA = 0.99 ms
	Printed Pages	OK	12-07-2011 05:35:04	0d 0h 5m 42s	1/3	Pagecount is 0
	Printer Consumables	OK	12-07-2011 05:35:19	0d 0h 5m 27s	1/3	Black Cartridge HP CE260A is at 100% - OK! Cyan Cartridge HP CE261A is at 100% - OK! Magenta Cartridge HP CE263A is at 100% - OK! Yellow Cartridge HP CE262A is at 100% - OK! Image Transfer Kit HP CE249A is at 100% - OK! Image Fuser Kit HP 110V-CE246A, 220V-CE247A is at 100% - OK!
	Printer Model	OK	12-07-2011 05:37:34	0d 0h 3m 12s	1/3	HP Color LaserJet CP4520 Series, Serial # JPBCC8C0RT
	Printer Status	OK	12-07-2011 05:36:50	0d 0h 3m 56s	1/3	Printer ok - ("Ready 192.168.5.12")
	Printer Tray 2	OK	12-07-2011 05:40:05	0d 0h 10m 41s	1/3	TRAY 2 is OK!

```
define service{
    use                  generic-service
    host_name           hp4300
    service_description Printer Consumables
    check_command       check_snmp_printer_args!-C public!-x "CONSUM ALL"
    normal_check_interval 10
    retry_check_interval 1
}
define service{
    use                  generic-service
    host_name           hp4300
    service_description Printer Tray 2
    check_command       check_snmp_printer_args!-C public!-x "TRAY 2"
```

```
normal_check_interval 10
retry_check_interval 1
}
define service{
    use                  generic-service
    host_name           hp4300
    service_description Printer Model
    check_command       check_snmp_printer_args!-C public!-x MODEL
    normal_check_interval 10
    retry_check_interval 1
}
define service{
    use                  generic-service
    host_name           hp4300
    service_description Printed Pages
    check_command       check_snmp_printer_args!-C public!-x PAGECOUNT
    normal_check_interval 10
    retry_check_interval 1
}
define service{
    use                  generic-service
    host_name           hp4300
    service_description Printer Messages
    check_command       check_snmp_printer_args!-C public!-x MESSAGES
    normal_check_interval 10
    retry_check_interval 1
}
```

Here is an example of an empty tray which sends a CRITICAL state.

Current Network Status

Last Updated: Fri Mar 11 03:55:08 MST 2011

Updated every 90 seconds

Nagios® Core™ 3.2.3 - www.nagios.org

Logged in as nagiosadmin

[View History For This Host](#)[View Notifications For This Host](#)[View Service Status Detail For All Hosts](#)**Host Status Totals**

Up Down Unreachable Pending

1 0 0 0

[All Problems](#) [All Types](#)

0 1

Service Status Totals

Ok Warning Unknown Critical Pending

4 0 0 1 0

[All Problems](#) [All Types](#)

1 5

Service Status Details For Host 'hp4300'

Host ↑↓	Service ↑↓	Status ↑↓	Last Check ↑↓	Duration ↑↓	Attempt ↑↓	Status Information
hp4300	PING	OK	03-11-2011 03:46:27	0d 1h 8m 41s	1/3	PING OK - Packet loss = 0%, RTA = 6.04 ms
	Printer Consumables	OK	03-11-2011 03:45:25	0d 0h 29m 43s	1/3	Black Print Cartridge HP Q1339A is at 46% - OK!
	Printer Model	OK	03-11-2011 03:48:43	0d 0h 16m 25s	1/3	Maintenance Kit HP 110V-Q2436A, 220V-Q2437A is at 26% - OK!
	Printer Status	OK	03-11-2011 03:48:01	0d 1h 7m 7s	1/3	hp LaserJet 4300, Serial # CNGY510590
	Printer Tray 2	CRITICAL	03-11-2011 03:52:02	0d 0h 5m 6s	3/3	TRAY 2 is at CRITICAL level - please refill with more 8.5 x 11.0 paper.

Chapter 7: Practical Exercises

The best way to learn anything with Nagios is to perform the task in a step-by-step manner so an understanding of the context and options of the situation help build a higher level of understanding. The Nagios Certified Professional will typically be engaged in recognizing and responding to problems discovered by Nagios as well as be responsible to create reports and locate decision making data about the network and devices on the network. Therefore, these exercises are designed to not only aid in preparing for the Nagios Certified Professional certification but also as a exercise for those working at a helpdesk situation.

Exercise #1: Login and Research

Management has requested that you log into the Nagios interface and provide for them research on a plugin that could be used to check for updates to a Linux server. The Linux servers are all running CentOS and management wants to make sure that updates are a priority for those machines. Several other tasks were requested as well:

- * verify that Nagios is the newest version
- * recommend several options for plugins to check on updates to a CentOS system
- * research the forum for ideas and problems related to the plugin choices

Login to Nagios

Point your browser to the IP Address of the Nagios server followed by “nagios”. Enter your username and password. Here is an example of the location in the browser.



Locate the installed version and then click the highlighted link to verify the latest version is installed.

Nagios® Core™ Version 3.3.1

July 25, 2011

[Check for updates](#)

Get Started

- Start monitoring your infrastructure
- Change the look and feel of Nagios
- Extend Nagios with hundreds of addons
- Get support
- Get training
- Get certified

Don't Miss...

- Improve your Nagios skillset with self-paced and instructor led [training services](#).
- The new [Nagios SNMP Trap Interface](#) project makes managing traps easier.
- Monitor business processes with the new [Nagios BPI](#) addon.

Quick Links

- Nagios Library (tutorials and docs)
- Nagios Labs (development blog)
- Nagios Exchange (plugins and addons)

Latest News

- Nagios Conference 2012 Call For Papers Opens
- Nagios Wins LinuxQuestions.org Network Monitoring Award By A Landslide!
- Last Chance - Vote For Nagios As Your Favorite Monitoring App
- More news...

Copyright © 2010-2012 Nagios Core Development Team and Community Contributors. Copyright © 1999-2009 Ethan Galstad. See the THANKS file for more information on contributors.

Nagios Core is licensed under the GNU General Public License and is provided AS IS with NO WARRANTY OF ANY KIND, INCLUDING THE WARRANTY OF DESIGN, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. Nagios, Nagios Core and the Nagios logo are trademarks, servicemarks, registered trademarks or registered servicemarks owned by Nagios Enterprises, LLC. Use of the Nagios marks is governed by the [trademark use restrictions](#).

Nagios®
MONITORED BY
NAGIOS CORE

SOURCEFORGE.NET

Once you have checked for to verify that you are using the latest version, then return to the “Home” page and review the links under “Get Started”, “Quick Links”, “Don’t Miss” and “latest News”. One of the tasks requested was research for a plugin so on the menu select “Nagios Exchange” under “Quick Links”.

Get Started

- Start monitoring your infrastructure
- Change the look and feel of Nagios
- Extend Nagios with hundreds of addons
- Get support
- Get training
- Get certified

Don't Miss...

- Improve your Nagios skillset with self-paced and instructor led [training services](#).
- The new [Nagios SNMP Trap Interface](#) project makes managing traps easier.
- Monitor business processes with the new [Nagios BPI](#) addon.

Quick Links

- Nagios Library (tutorials and docs)
- Nagios Labs (development blog)
- Nagios Exchange (plugins and addons)**

Latest News

- Nagios Conference 2012 Call For Papers Opens
- Nagios Wins LinuxQuestions.org Network Monitoring Award By A Landslide!
- Last Chance - Vote For Nagios As Your Favorite Monitoring App
- More news...

Follow the link to <http://exchange.nagios.org/> and then select plugins to begin your search.

The screenshot shows the Nagios Exchange homepage. At the top, there's a navigation bar with links for Home, Directory, and About. Below the navigation is a section titled "Nagios Exchange" which includes a brief description of the site as a central place for sharing Nagios projects and a call to action to share new projects. To the right of this text is a graphic of six orange human figures connected by lines, forming a network around a central circle labeled "Nagios EXCHANGE".

Featured Project:

Monitoring Hosts Using NRPE
★★★★★
 /Category: Nagios XI Documentation
 Views: 20148

Project Categories:

- [Addons](#) (522)
- [Certified Compatible](#) (3)
- [Comparisons](#) (8)
- [Cool Stuff](#) (6)
- [Demos](#) (3)
- [Distributions](#) (18)
- [Documentation](#) (120)
- [Graphics and Logos](#) (35)
- [Media Coverage](#) (5)
- [Multimedia](#) (102)
- [Patches](#) (19)
- [Plugins](#) (2478) (2478)
- [Seedcamp](#) (14)
- [Translations](#) (8)
- [Tutorials](#) (174)
- [Uncategorized](#) (0)
- [Utilities](#) (14)

Popular Projects:

Jump to the most requested...

Nagios XI Addons:

- [Config Wizards](#)
- [Components](#)

General Addons:

- [Nagios BPI](#)
- [Exfoliation](#)
- [nagiosgraph](#)
- [Nagios V-Shell](#)
- [NRDP](#)
- [NRPE](#)
- [NSClient++](#)
- [Nageventlog](#)

Once you get to the next page select “Operating Systems” and then “Linux” and start reviewing the plugins for checking for updates. You will eventually get to a plugin that is highly rated, note the stars and easy to implement.

Check_Yum ★★★★★

harisekhon.com/nagios

Checks for Yum Updates on RedHat/CentOS systems. Shows the number of updates and can differentiate between security and regular updates. Switches to control alerting behaviour as well as whether to update the cache or enable/disable repositories

From all indications this would be the plugin to recommend. Before you make that decision return to the “Home” for Nagios and again use the menu to select the forum that is available for help.

Get Started

- Start monitoring your infrastructure
- Change the look and feel of Nagios
- Extend Nagios with hundreds of addons
- Get support
- Get training
- Get certified

Don't Miss...



- Improve your Nagios skillset with self-paced and instructor led [training services](#).
- The new [Nagios SNMP Trap Interface](#) project makes managing traps easier.
- Monitor business processes with the new [Nagios BPI](#) addon.

Quick Links

- [Nagios Library](#) (tutorials and docs)
- [Nagios Labs](#) (development blog)
- [Nagios Exchange](#) (plugins and addons)

Latest News

- Nagios Conference 2012 Call For Papers Opens
- Nagios Wins LinuxQuestions.org Network Monitoring Award By A Landslide!
- Last Chance - Vote For Nagios As Your Favorite Monitoring App
- [More news...](#)

Once you get to the “Support Resources” page select “General Support Forum” as your company has not purchased a support package.

Network: | Enterprise | [Support](#) | Library | Project | Exchange | [+]



Support

Where You Get Your Fix™

[Home](#) [Knowledge Base](#) [Learn More](#) [Get Started](#) [Contact Support](#)

Nagios Support Resources

Find the most up-to-date support information on Nagios by utilizing the resources below.

Support Forum

Get commercial and community help on the official Nagios support forum at [support.nagios.com/forum](#)

Quick Links:

- [Customer Support Forum](#)
- [General Support Forum](#)

Documentation

Official manuals, documentation, and video tutorials for Nagios can be found below.

- **Nagios XI:**
 - [Administrator Guide](#)
 - [User Guide](#)
 - [Documentation and HOWTOs](#)
 - [Video Tutorials](#)
 - [FAQs](#)
- **Nagios Core:**
 - [User Guide](#)
 - [Documentation and HOWTOs](#)
 - [Video Tutorials](#)
 - [FAQs](#)

Nagios Library

Additional documentation and video tutorials for Nagios is available at the [Nagios Library](#) at [library.nagios.com](#).

Search...

Download Nagios

Annual Support Contracts

Learn more about annual Nagios support plans starting at just \$1,295 USD.

Stay Informed

[Subscribe To Our Newsletter](#)

Privacy by  SafeSubscribe™

Copyright by Nagios Enterprises, LLC
Cannot be reproduced without written permission. P.O. Box 8154, Saint Paul, MN 55108

Once at the support forum choose the “Nagios Core” forum and then search to see if there are any problems using the check_yum plugin that you located.

The screenshot shows the Nagios Support Forum interface. At the top, the Nagios logo and the text "Nagios® Support Forum" are displayed, along with a search bar and links for "Advanced search", "FAQ", "Members", and "Logout". Below the header, the "General Support" board index is shown. It includes a breadcrumb trail ("Board index < General Support"), a link to "User Control Panel (0 new messages) • View your posts", and a "Mark forums read" link. The main content area lists various forums with their names, descriptions, topic counts, post counts, and last post details. The forums listed are:

FORUM	TOPICS	POSTS	LAST POST
Nagios XI This board serves as an open discussion and support collaboration point for Nagios XI . NOTE: Nagios XI customers should use the Customer Support forum to obtain expedited support.	1474	8177	by scottwilkerson Wed Mar 21, 2012 10:05 am
Nagios Fusion This board serves as an open discussion and support collaboration point for Nagios Fusion . NOTE: Nagios Fusion customers should use the Customer Support forum to obtain expedited support.	18	107	by mguthrie Wed Mar 21, 2012 10:46 am
Nagios Core An open discussion forum for obtaining help with Nagios Core.	1214	3447	by darthcolo Wed Mar 21, 2012 9:17 am
Nagios V-Shell Information and discussions relating to the Nagios V-Shell web interface.	27	120	by mguthrie Tue Mar 13, 2012 10:03 am
Nagios SNMP Trap Interface Information and discussions relating to the NSTI (Nagios SNMP Trap Interface) addon.	8	25	by momzic Wed Mar 14, 2012 7:45 am
Nagios Mobile Information and discussions relating to the Nagios Mobile addon.	5	16	by mguthrie Wed Mar 14, 2012 12:41 pm
Nagios Demo VM Information and discussions about the Nagios demo virtual machine (VM). More information on the Nagios Demo VM can be found at go.nagios.com/demovm	8	25	by mguthrie Thu Feb 09, 2012 1:24 pm
Nagios DVD Information and discussions pertaining to the Nagios DVD. Additional information about the Nagios DVD can be found online at go.nagios.com/dvd	5	18	by mguthrie Fri Sep 23, 2011 8:37 am

These steps allow you to fulfill the request made by management.

Exercise #2: Responding to Problems

You have been notified of two problems on the network. The first problem is that a host is down, however you have since been notified that someone is working on that problem. The second problem is that a service is flapping and you need to disable the flap detection for that service. Third, you were told that a service is now up but you want to reschedule a check to be performed immediately to verify.

Proceed to the “home” page and on the menu select “Hosts(unhandled)” as this will list the hosts that are not currently responding to connections from Nagios.

Here you can see the “cisco827” is down. Click on the host.

Host	Status	Last Check	Duration	Status Information
cisco827	DOWN	03-21-2012 11:59:30	6d 19h 44m 14s	CRITICAL - Host Unreachable (192.168.5.228)

Once you have selected the host, now select “Acknowledge this host problem” as you want to stop notifications and communicate to the rest of those watching Nagios.

Host Information

Last Updated: Wed Mar 21 12:20:32 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

View Status Detail For This Host
 View Alert History For This Host
 View Trends For This Host
 View Alert Histogram For This Host
 View Availability Report For This Host
 View Notifications For This Host

Host
Cisco router
(cisco827)

Member of
No hostgroups

192.168.5.228

Host State Information		Host Commands
Host Status:	DOWN (for 6d 20h 3m 32s)	
Status Information:	CRITICAL - Host Unreachable (192.168.5.228)	
Performance Data:		
Current Attempt:	1/10 (HARD state)	
Last Check Time:	03-21-2012 12:20:10	
Check Type:	ACTIVE	
Check Latency / Duration:	0.118 / 3.007 seconds	
Next Scheduled Active Check:	03-21-2012 12:25:20	
Last State Change:	03-14-2012 16:17:00	
Last Notification:	03-21-2012 11:54:50 (notification 328)	
Is This Host Flapping?	NO (0.00% state change)	
In Scheduled Downtime?	NO	
Last Update:	03-21-2012 12:20:30 (0d 0h 0m 2s ago)	
Active Checks:	ENABLED	
Passive Checks:	ENABLED	
Obsessing:	ENABLED	
Notifications:	ENABLED	
Event Handler:	ENABLED	
Flap Detection:	ENABLED	

Host Comments

Add a new comment Delete all comments

Entry Time	Author	Comment	CommentID	Persistent	Type	Expires	Actions
This host has no comments associated with it							

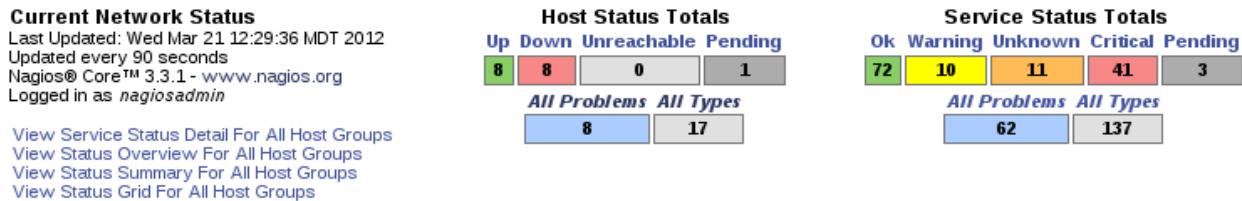
Now you should enter the information that you want to relate to other administrators. In this case the IOS is being updated so the router is down.

Command Options

Host Name:	<input type="text" value="cisco827"/>
Sticky Acknowledgement:	<input checked="" type="checkbox"/>
Send Notification:	<input checked="" type="checkbox"/>
Persistent Comment:	<input type="checkbox"/>
Author (Your Name):	<input type="text" value="Nagios Admin"/>
Comment:	<input type="text" value="New IOS being tested for this router."/>
Commit Reset	

Once you hit “Commit” this information will now be made available on the web site so others know the problem is taken is being handled. Now when you access the host you can see the check mark indicating someone has

acknowledged the problem and the comment icon tells them there is a text comment telling them more information.



Display Filters:
Host Status Types: All problems
Host Properties: Any
Service Status Types: All
Service Properties: Any

Host Status Details For All Host Groups

Host	Status	Last Check	Duration	Status Information
cisco827	DOWN	03-21-2012 12:25:20	6d 20h 12m 36s	CRITICAL - Host Unreachable (192.168.5.228)

Now you need to access a service that is stable currently but has a history of flapping. Management has requested that you turn flapping off for this service. Choose the service and select “Disable flap detections for this service”.

Service Information

Last Updated: Wed Mar 21, 12:53:44 MDT 2012
Updated every 90 seconds
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as nagiosadmin

Service
Mailq
On Host
Company Server
(bash)

Member of
No servicegroups.

192.168.5.190

Extra Actions

Service State Information

Current Status:	OK (for 70d 7h 41m 22s)
Status Information:	OK: mailq reports queue is empty
Performance Data:	unsent=0;5;10;0
Current Attempt:	1/3 (HARD state)
Last Check Time:	03-21-2012 12:53:20
Check Type:	ACTIVE
Check Latency / Duration:	0.107 / 0.065 seconds
Next Scheduled Check:	03-21-2012 13:03:20
Last State Change:	01-11-2012 04:12:22
Last Notification:	N/A (notification 0)
Is This Service Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	03-21-2012 12:53:40 (0d 0h 0m 4s ago)

Service Commands

- Disable active checks of this service
- Re-schedule the next check of this service
- Submit passive check result for this service
- Stop accepting passive checks for this service
- Stop obsessing over this service
- Disable notifications for this service
- Send custom service notification
- Schedule downtime for this service
- Disable event handler for this service
- Disable flap detection for this service

Service Comments

Add a new comment Delete all comments

Entry Time	Author	Comment	CommentID	Persistent	Type	Expires	Actions
This service has no comments associated with it							

Now when you access the service the flap detection is turned off and it is listed on the service so other administrators know as well. It can be turned back on at any time.

Service State Information		Service Commands
Current Status:	OK (for 70d 7h 45m 15s)	Disable active checks of this service
Status Information:	OK: mailq reports queue is empty	Re-schedule the next check of this service
Performance Data:	unsent=0;5;10;0	Submit passive check result for this service
Current Attempt:	1/3 (HARD state)	Stop accepting passive checks for this service
Last Check Time:	03-21-2012 12:53:20	Stop obsessing over this service
Check Type:	ACTIVE	Disable notifications for this service
Check Latency / Duration:	0.107 / 0.065 seconds	Send custom service notification
Next Scheduled Check:	03-21-2012 13:03:20	Schedule downtime for this service
Last State Change:	01-11-2012 04:12:22	Disable event handler for this service
Last Notification:	N/A (notification 0)	Enable flap detection for this service
Is This Service Flapping?	N/A	
In Scheduled Downtime?	NO	
Last Update:	03-21-2012 12:57:30 (0d 0h 0m 7s ago)	
Active Checks:	ENABLED	
Passive Checks:	ENABLED	
Obsessing:	ENABLED	
Notifications:	ENABLED	
Event Handler:	ENABLED	
Flap Detection:	DISABLED	

The final task is to recheck a service to make sure it is up. The first thing you do is check the scheduling to see where the check is. Proceed to the menu and “System” and then “scheduling Queue”. As you review the queue you see that your check will not happen for some time and you do not want to wait for it.

Check Scheduling Queue
 Last Updated: Wed Mar 21 13:08:16 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Entries sorted by **next check time** (ascending)

Host ↑↓	Service ↑↓	Last Check ↑↓	Next Check ↑↓	Type	Active Checks	Actions
vz		03-21-2012 13:03:00	03-21-2012 13:08:10	Normal	ENABLED	
ub3		03-21-2012 13:03:00	03-21-2012 13:08:10	Normal	ENABLED	
ub1		03-21-2012 13:03:00	03-21-2012 13:08:10	Normal	ENABLED	
ub		03-21-2012 13:03:00	03-21-2012 13:08:10	Normal	ENABLED	

Since your check will not come up for a while, go to the menu and select the service you want to verify. Once you open the service select “Re-schedule the next check of this service”.

Service Information

Last Updated: Wed Mar 21 13:12:04 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

[View Information For This Host](#)
[View Status Detail For This Host](#)
[View Alert History For This Service](#)
[View Trends For This Service](#)
[View Alert Histogram For This Service](#)
[View Availability Report For This Service](#)
[View Notifications For This Service](#)

Service
Web Server
 On Host
Ubuntu Server
(ub1)

Member of
No servicegroups.

192.168.5.181

Service State Information		Service Commands
Current Status:	OK (for 132d 0h 49m 12s)	Disable active checks of this service
Status Information:	TCP OK - 0.000 second response time on port 80	Pe-schedule the next check of this service
Performance Data:	time=0.000106s;0.000000;10.000000	Submit passive check result for this service
Current Attempt:	1/3 (HARD state)	Stop accepting passive checks for this service
Last Check Time:	03-21-2012 13:06:36	Stop obsessing over this service
Check Type:	ACTIVE	Disable notifications for this service
Check Latency /Duration:	0.110 / 0.011 seconds	Send custom service notification
Next Scheduled Check:	03-21-2012 13:16:36	Schedule downtime for this service
Last State Change:	11-10-2011 11:22:52	Disable event handler for this service
Last Notification:	N/A (notification 0)	Disable flap detection for this service
Is This Service Flapping?	NO (0.00% state change)	
In Scheduled Downtime?	NO	
Last Update:	03-21-2012 13:12:00 (0d 0h 0m 4s ago)	
Active Checks:	ENABLED	
Passive Checks:	ENABLED	
Obsessing:	ENABLED	
Notifications:	ENABLED	
Event Handler:	ENABLED	
Flap Detection:	ENABLED	

Once you click it you will need to “Commit”. Note, you could set the time specifically. The force option will check immediately.

You are requesting to schedule a service check

Command Options		Command Description
Host Name:	ub1	This command is used to schedule the next check of a particular service. Nagios will re-queue the service to be checked at the time you specify. If you select the <i>force check</i> option, Nagios will force a check of the service regardless of both what time the scheduled check occurs and whether or not checks are enabled for the service.
Service:	Web Server	
Check Time:	03-21-2012 13:16:01	
Force Check:	<input checked="" type="checkbox"/>	
Commit Reset		

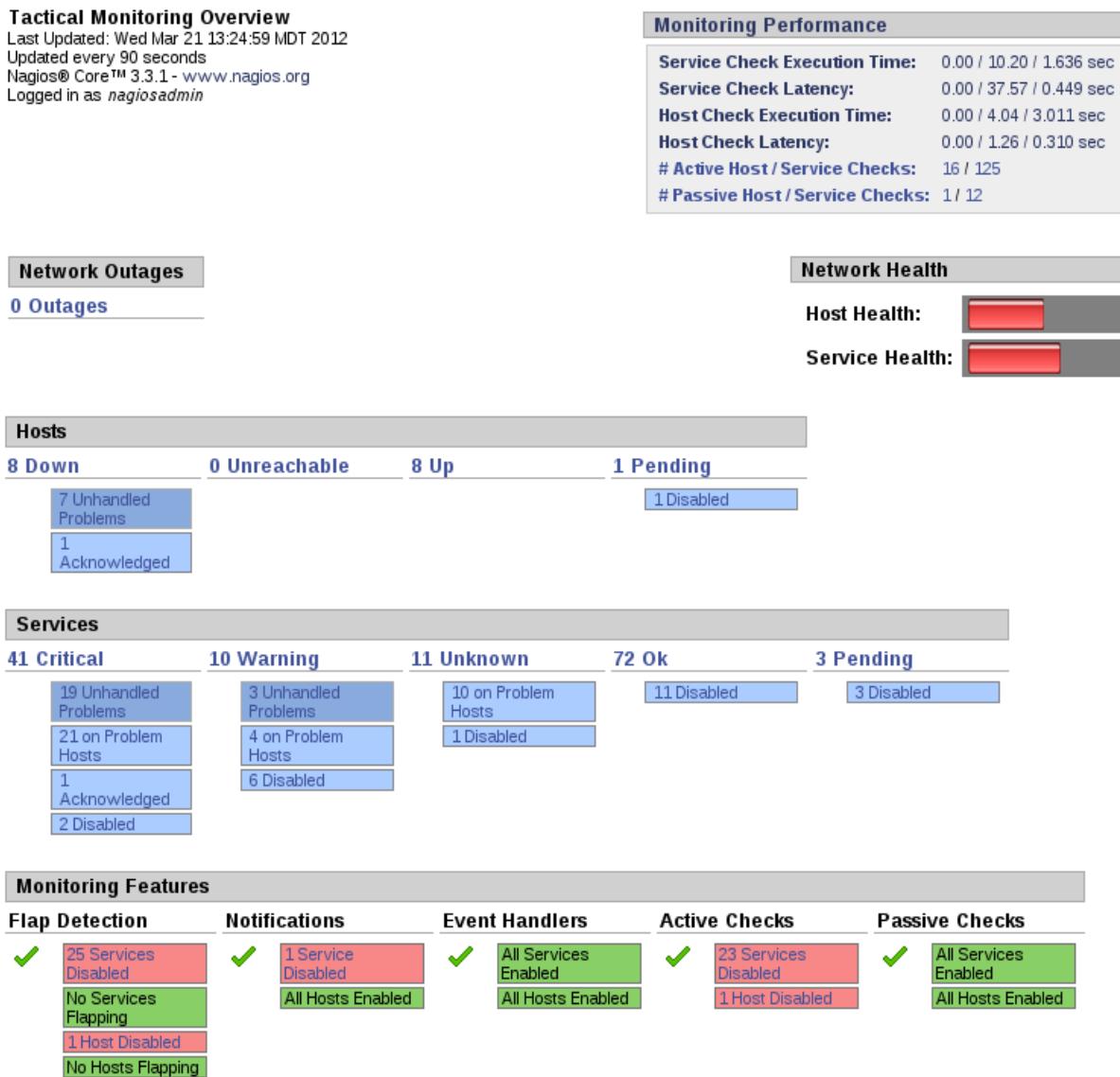
That completes all of the tasks in this exercise.

Exercise #3: Reports

Management has requested a number of reports to verify activity on the network. Here is the list of reports that they would like to see:

- * overview of the entire system and how it is running currently
- * summary of all host and service alerts over the last week
- * uptime of a specific host as it is mission critical
- * a graph of alerts for a specific host
- * latency report for hosts and services

The organization has a power outage and once the power is back on the management asks you to show them the current status of the network. Proceed to the menu and “Current Status” and “Tactical Overview”.



This snapshot of the network not only shows the status of hosts and services but also the features that you are using on the network, like flapping, notifications, eventhandlers, etc. Notice the “Monitoring Performance” provides latency information, both for services and hosts.

In order to get a different picture on latency open up “System” and “Performance Info”.

Performance Information

Last Updated: Thu Mar 22 06:50:54 MDT 2012
 Updated every 90 seconds
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Services Actively Checked:

Time Frame	Services Checked
<= 1 minute:	9 (7.6%)
<= 5 minutes:	60 (50.4%)
<= 15 minutes:	116 (97.5%)
<= 1 hour:	116 (97.5%)
Since program start:	116 (97.5%)

Services Passively Checked:

Time Frame	Services Checked
<= 1 minute:	3 (16.7%)
<= 5 minutes:	12 (66.7%)
<= 15 minutes:	16 (88.9%)
<= 1 hour:	16 (88.9%)
Since program start:	18 (100.0%)

Hosts Actively Checked:

Time Frame	Hosts Checked
<= 1 minute:	4 (25.0%)
<= 5 minutes:	13 (81.2%)
<= 15 minutes:	15 (93.8%)
<= 1 hour:	15 (93.8%)
Since program start:	15 (93.8%)

Hosts Passively Checked:

Time Frame	Hosts Checked
<= 1 minute:	0 (0.0%)
<= 5 minutes:	1 (100.0%)
<= 15 minutes:	1 (100.0%)
<= 1 hour:	1 (100.0%)
Since program start:	1 (100.0%)

Check Statistics:

Type	Last 1 Min	Last 5 Min	Last 15 Min
Active Scheduled Host Checks	1	14	45
Active On-Demand Host Checks	4	18	51
Parallel Host Checks	2	16	49
Serial Host Checks	0	0	0
Cached Host Checks	3	17	47
Passive Host Checks	0	3	9
Active Scheduled Service Checks	11	67	204
Active On-Demand Service Checks	0	0	0
Cached Service Checks	0	0	0
Passive Service Checks	4	13	38
External Commands	4	17	50

Buffer Usage:

Type	In Use	Max Used	Total Available
External Commands	0	9	4096

The advantage is that there is more information available and it is easier to understand. For example, the “Tactical

Overview” provides these metrics for service latency:

0.00/37.57/0.449 sec.

Of course this is minimum/maximum/average. So it will provide a quick view of latency. However, the “Performance Info” page provides that information as well as check time and percentage of change. The percentage of change can be a way to predict problems before they actually occur.

The next aspect that management wants to see are all alerts for hosts and services over the last week. When you click on “Reports” and “Alert Summary” you can set the time range for the list. In fact, there are a lot of options that can be created for exactly the type of report needed.

Alert Summary Report

Last Updated: Thu Mar 22 07:03:36 MDT 2012
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Standard Reports:

Report Type:

Custom Report Options:

Report Type:

Report Period:

If Custom Report Period...

Start Date (Inclusive):

End Date (Inclusive):

Limit To Hostgroup:

Limit To Servicegroup:

Limit To Host:

Alert Types:

State Types:

Host States:

Service States:

Max List Items:

The report itself can be helpful not only in recognition of problems that have occurred but it can also be useful in predicting future problems.

Alert Summary Report						
Last Updated: Thu Mar 22 07:05:31 MDT 2012 Nagios® Core™ 3.3.1 - www.nagios.org Logged in as <i>nagiosadmin</i>						
Most Recent Alerts						
03-15-2012 07:05:31 to 03-22-2012 07:05:31 Duration: 7d 0h 0m 0s						
Report Options Summary:						
Alert Types: Host & Service Alerts State Types: Soft & Hard States Host States: Up, Down, Unreachable Service States: Ok, Warning, Unknown, Critical						
Generate New Report						
Displaying most recent 25 of 7026 total matching alerts						
Time	Alert Type	Host	Service	State	State Type	Information
03-22-2012 07:04:42	Service Alert	google	Weblinject Google Search	OK	SOFT	Weblinject OK - All tests passed successfully in 0.357 seconds
03-22-2012 07:03:42	Service Alert	google	Weblinject Google Search	CRITICAL	SOFT	(Return code of 255 is out of bounds)
03-22-2012 07:02:52	Service Alert	cisco_catalyst	Port 1 Bandwidth Usage	OK	HARD	Traffic OK - Avg. In = 0.0 B/s, Avg. Out = 0.0 B/s
03-22-2012 07:01:32	Service Alert	localhost	HTTP	OK	SOFT	HTTP OK: HTTP/1.1 200 OK - 328 bytes in 0.000 second response time
03-22-2012 07:00:32	Service Alert	localhost	HTTP	CRITICAL	SOFT	(Return code of 127 is out of bounds - plugin may be missing)
03-22-2012 06:56:32	Service Alert	localhost	HTTP	OK	SOFT	HTTP OK: HTTP/1.1 200 OK - 328 bytes in 0.000 second response time
03-22-2012 06:55:32	Service Alert	localhost	HTTP	CRITICAL	SOFT	(Return code of 127 is out of bounds - plugin may be missing)
03-22-2012 06:54:42	Service Alert	google	Weblinject Google Search	OK	SOFT	Weblinject OK - All tests passed successfully in 0.358 seconds

Management would also like to have a look at a graph showing the alerts for a specific host that is a mission critical host. Once clicking on “Reports/Alerts Histogram” you can make the decision about using the report for a host or a device.

Host and Service Alert Histogram
 Last Updated: Thu Mar 22 07:18:37 MDT 2012
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Step 1: Select Report Type

Type: **Host** ▾

Continue to Step 2

Select the host from the list on the next screen and then select the time frame that you want to see.

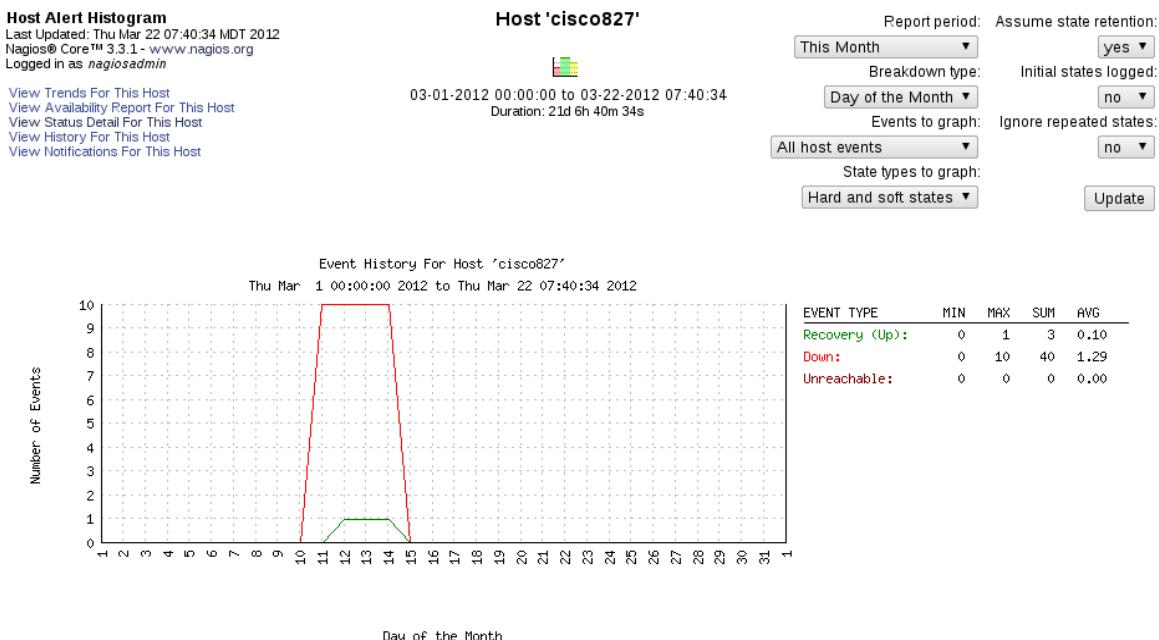
Host Alert Histogram

Last Updated: Thu Mar 22 07:43:33 MDT 2012
 Nagios® Core™ 3.3.1 - www.nagios.org
 Logged in as *nagiosadmin*

Step 3: Select Report Options

Report Period: <input style="width: 150px; height: 20px;" type="button" value="This Month"/> If Custom Report Period... Start Date (Inclusive): <input style="width: 150px; height: 20px;" type="button" value="March 1 2012"/> End Date (Inclusive): <input style="width: 150px; height: 20px;" type="button" value="March 22 2012"/> Statistics Breakdown: <input style="width: 150px; height: 20px;" type="button" value="Day of the Month"/> Events To Graph: <input style="width: 150px; height: 20px;" type="button" value="All host events"/> State Types To Graph: <input style="width: 150px; height: 20px;" type="button" value="Hard and soft states"/> Assume State Retention: <input style="width: 150px; height: 20px;" type="button" value="Yes"/> Initial States Logged: <input style="width: 150px; height: 20px;" type="button" value="No"/> Ignore Repeated States: <input style="width: 150px; height: 20px;" type="button" value="No"/> <input style="width: 150px; height: 20px;" type="button" value="Create Report"/>

This report demonstrates a history of the problems related to this host. It shows the time it was down and the recovery state.



Now management needs to see uptime on a host, in other words how available was this host for service over the last 30

days. Proceed to “Reports” and “Availability” and then select the host option.

Availability Report

Last Updated: Thu Mar 22 08:06:48 MDT 2012
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 1: Select Report Type

Type:

Select the host from your list of hosts.

Availability Report

Last Updated: Thu Mar 22 08:12:29 MDT 2012
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 2: Select Host

Host(s):

*Tip: If you want to have the option of getting the availability data in CSV format, select ** ALL HOSTS ** from the pull-down menu.*

Once the host is selected choose the options that you want to see, in this example the availability of the host for the last 31 days is selected.

Host Availability Report
Last Updated: Thu Mar 22 08:14:02 MDT 2012
Nagios® Core™ 3.3.1 - www.nagios.org
Logged in as *nagiosadmin*

Step 3: Select Report Options

Report Period: ▼

If Custom Report Period...

Start Date (Inclusive): ▼

End Date (Inclusive): ▼

Report time Period: ▼

Assume Initial States: ▼

Assume State Retention: ▼

Assume States During Program Downtime: ▼

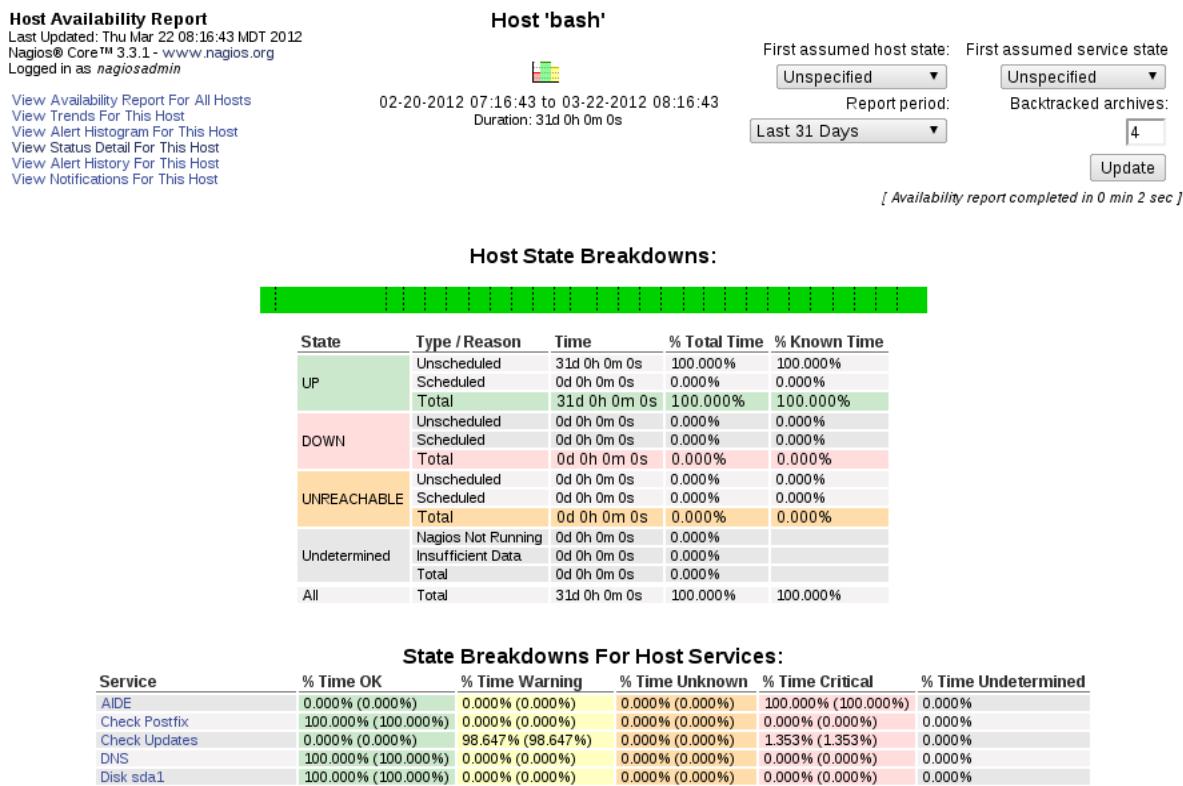
Include Soft States: ▼

First Assumed Host State: ▼

First Assumed Service State: ▼

Backtracked Archives (To Scan For Initial States):

The report provides a graph at the top which is green/red based on availability and then provides detailed information below that. The availability of services are also listed on the chart.



This completes all of the tasks for this exercise.

Exercise #4: Passive vs. Active Checks

The goal in this exercise is to recognize the difference in viewing active checks versus viewing passive checks because they must be interpreted correctly. Your organization has a lot of active checks but it also has some passive checks and you need to be able to understand the differences. First, remember that active checks are when Nagios takes the initiative to execute the timetable for the check while passive checks are implemented solely on the client. Here is an example of both active and passive checks. The active checks are NRPE, NTP and Nagios DNS. These three checks are initiated by the Nagios server and the data is returned to Nagios.

NRPE	OK	03-22-2012 08:53:20	71d 3h 43m 18s	1/3	NRPE v2.12
NTP	OK	03-22-2012 08:48:56	12d 17h 39m 52s	1/3	NTP OK: Offset 3.952945744 secs
Nagios DNS	OK	03-22-2012 08:56:36	30d 5h 37m 28s	1/3	DNS OK: 0.070 seconds response time. nagios.com returns 173.45.234.224
Pass-BasicUsers	?	WARNING	03-22-2012 08:56:09	48d 20h 18m 12s	3/3
Pass-HostStatus	?	WARNING	03-22-2012 08:56:09	48d 20h 18m 12s	3/3
Pass-Processes	?	WARNING	03-22-2012 08:56:09	48d 20h 18m 12s	3/3
Pass-Users	?	PENDING	N/A	10d 20h 16m 38s+	1/3
Pass-WebAttack	?	PENDING	N/A	10d 20h 16m 38s+	1/3
Pass-WebMultiAttack	?	PENDING	N/A	10d 20h 16m 38s+	1/3

You can also see the passive checks in two different situations. First, recognize that the “?” is the icon used in this frontend, exfoliation (the default for Nagios Core), to indicate it is a passive check. So the first thing you know is all

passive checks will have the icon representing the fact that it is passive. Those passive checks listed which are in a WARNING state also indicate that a check has not been received from the client for over an hour. These three have an additional script tied to them which will provide a WARNING if the client has not sent a check for that service. Again, the importance here is that the client initiates the check so that if for any reason the client does not send a check the state of the passive check WILL NOT CHANGE. This is critical to understand and that is why the check has a time limit set so the administrator understands that the check has not been received.

The other three passive checks which are “PENDING” indicate that a check has never been received. Again, if the client does not send the checks, Nagios cannot effectively monitor that client.

Here is an example of one active check, SNMP Memory Usage, and three passive checks, again note the “?”. Here the story is different in that the client has sent passive check information for these checks to Nagios.

SNMP Memory usage	OK	03-22-2012 08:53:26	71d 3h 48m 2s	1/3	Ram : 33%, Swap : 0% :: OK
SSH Check Disk	?	03-22-2012 08:57:33	16d 14h 12m 20s	1/3	DISK OK - free space: / 1136 MB (37% inode=79%)
SSH Check Load	?	03-22-2012 08:56:43	16d 14h 12m 20s	1/3	OK - load average: 0.00, 0.00, 0.00
SSH Check Users	?	03-22-2012 08:57:53	16d 14h 12m 20s	1/3	USERS OK - 0 users currently logged in

In any evaluation of the checks that are being used on the network, the administrator must first discern if the check is active or passive and then determine if the passive check is actually current. Discipline yourself to verify that the passive check is current or you can incorrectly make the assumption about the real state of the passive check.