Network Management & Monitoring

NAGIOS

Introduction

Network Monitoring Tools

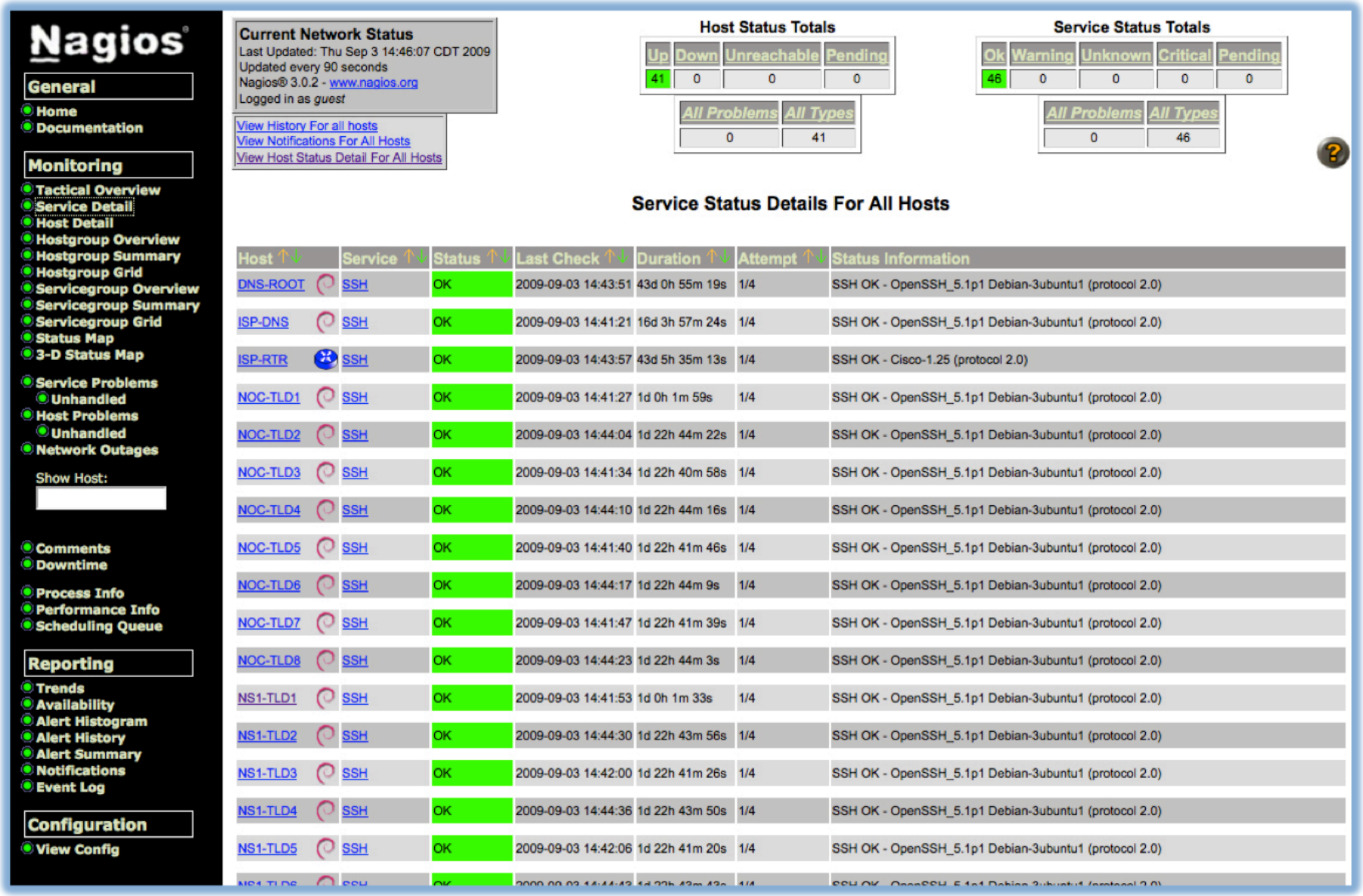
- Availability
- Reliability
- Performance

Nagios actively monitors the availability of devices and services

Introduction

- Possibly the most used open source network monitoring software
- Web interface for viewing status, browsing history, scheduling downtime etc
- Sends out alerts via E-mail. Can be configured to use other mechanisms, e.g. SMS

Example: Service Detail view



Features

Utilizes topology to determine dependencies.

- Differentiates between what is *down* vs. what is *unreachable*. Avoids running unnecessary checks and sending redundant alarms

Allows you to define how to send notifications based on combinations of:

- Contacts and lists of contacts
- Devices and groups of devices
- Services and groups of services
- Defined hours by persons or groups.
- The state of a service.

Plugins

Plugins are used to verify services and devices:

- Nagios architecture is simple enough that writing new plugins is fairly easy in the language of your choice.
- There are ***many, many*** plugins available (thousands).
  - ✓ http://exchange.nagios.org/
  - ✓ http://nagiosplugins.org/

Pre-installed plugins in Ubuntu

/usr/lib/nagios/plugins

check_apt	check_file_age	check_jabber	check_nntp	check_procs	check_swap
check_bgpstate	check_flexlm	check_ldap	check_nntps	check_radius	check_tcp
check_breeze	check_ftp	check_ldaps	check_nt	check_real	check_time
check_by_ssh	check_host	check_linux_raid	check_ntp	check_rpc	check_udp
check_clamd	check_hpjd	check_load	check_ntp_peer	check_rta_multi	check_ups
check_cluster	check_http	check_log	check_ntp_time	check_sensors	check_users
check_dhcp	check_icmp	check_mailq	check_nwstat	check_simap	check_wave
check_dig	check_ide_smart	check_mrtg	check_oracle	check_smtp	negate
check_disk	check_ifoperstatus	check_mrtgtraf	check_overcr	check_snmp	urlize
check_disk_smb	check_ifstatus	check_mysql	check_pgsql	check_spop	utils.pm
check_dns	check_imap	check_mysql_query	check_ping	check_ssh	utils.sh
check_dummy	check_ircd	check_nagios	check_pop	check_ssmtp	

/etc/nagios-plugins/config

apt.cfg	disk-smb.cfg	ftp.cfg	ldap.cfg	mysql.cfg	ntp.cfg	radius.cfg	ssh.cfg
breeze.cfg	dns.cfg	hppjd.cfg	load.cfg	netware.cfg	pgsql.cfg	real.cfg	tcp_udp.cfg
dhcp.cfg	dummy.cfg	http.cfg	mail.cfg	news.cfg	ping.cfg	rpc-nfs.cfg	telnet.cfg
disk.cfg	flexlm.cfg	ifstatus.cfg	mrtg.cfg	nt.cfg	procs.cfg	snmp.cfg	users.cfg

How checks work

- Periodically Nagios calls a plugin to test the state of each service. Possible responses are:
  - OK
  - WARNING
  - CRITICAL
  - UNKNOWN
- If a service is not OK it goes into a “soft” error state. After a number of retries (default 3) it goes into a “hard” error state. At that point an alert is sent.
- You can also trigger external event handlers based on these state transitions

How checks work continued

Parameters

- Normal checking interval
- Retry interval (i.e. when not OK)
- Maximum number of retries
- Time period for performing checks
- Time period for sending notifications

Scheduling

- Nagios spreads its checks throughout the time period to even out the workload
- Web UI shows when next check is scheduled

The concept of “parents”

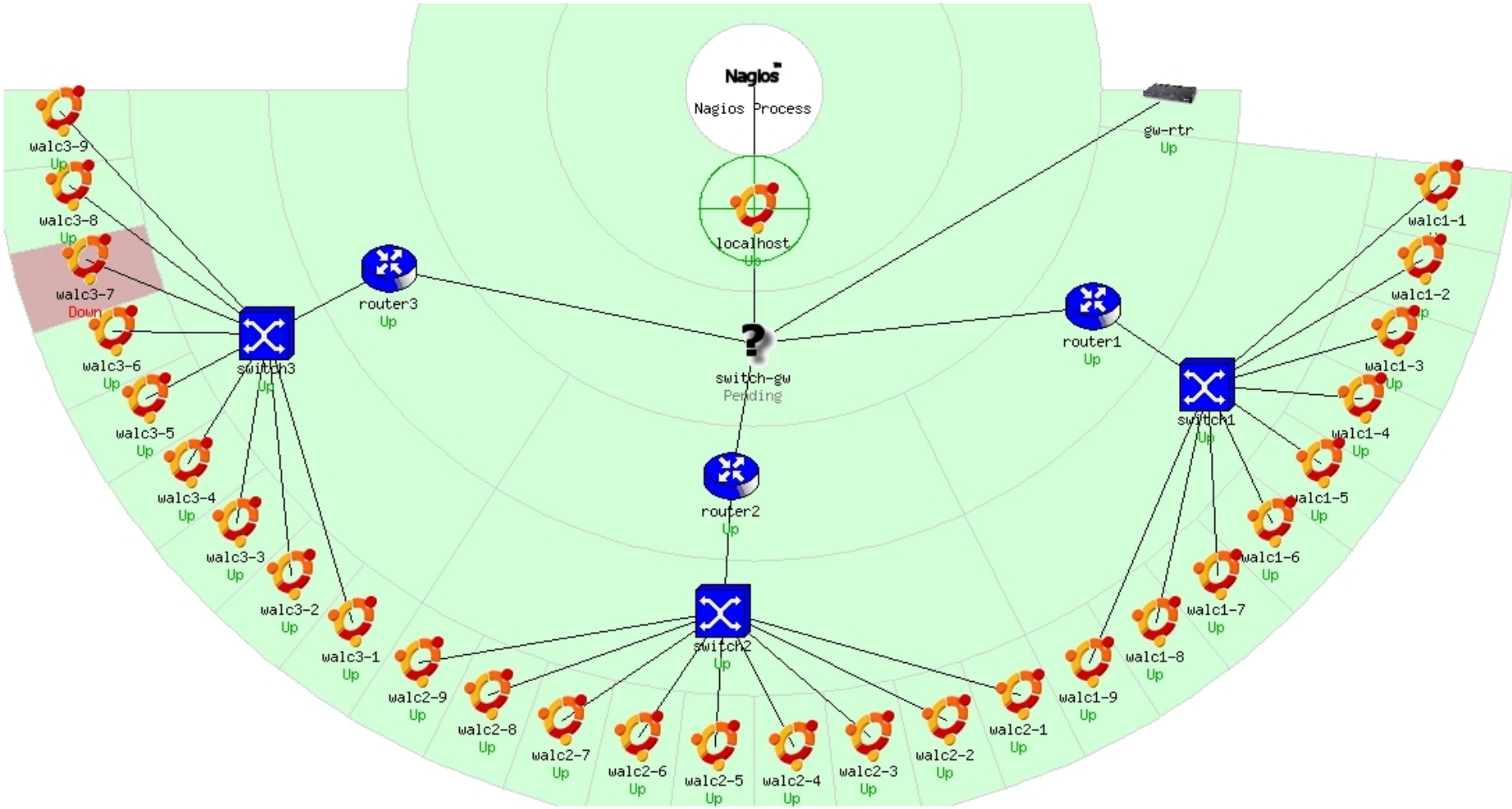
Hosts can have parents:

- The parent of a PC connected to a switch would be the switch.
- Allows us to specify the dependencies between devices.
- Avoids sending alarms when parent does not respond.
- A node can have multiple parents (dual homed).

Network viewpoint

- Where you locate your Nagios server will determine your point of view of the network.
- The Nagios server becomes the “root” of your dependency tree

Network viewpoint



Demo Nagios

Installation

In Debian/Ubuntu

```
# apt-get install nagios3
```

Key directories

```
/etc/nagios3
/etc/nagios3/conf.d
/etc/nagios-plugins/config
/usr/lib/nagios/plugins
/usr/share/nagios3/htdocs/images/logos
```

Nagios web interface is here:

http://pcN.ws.nsrc.org/nagios3/

Configuration

- Configuration defined in text files
  - /etc/nagios3/conf.d/*.cfg
  - Details at http://nagios.sourceforge.net/docs/3_0/objectdefinitions.html
- The default config is broken into several files with different objects in different files, but actually you can organise it how you like
- Always verify before restarting Nagios – otherwise your monitoring system may die!
  - `nagios3 –v /etc/nagios3/nagios.cfg`

Hosts and services configuration

Based on templates

- This saves lots of time avoiding repetition

There are default templates with default parameters for a:

- *generic host* (generic-host_nagios2.cfg)
- *generic service* (generic-service_nagios2.cfg)

- Individual settings can be overridden
- Defaults are all sensible

Monitoring a single host

pcs.cfg

```
define host {
    host_name pc1
    alias     pc1 in group 1
    address   pc1.ws.nsrc.org
    use       generic-host
}
```

copy settings from this template

- This is a minimal working config
  - You are just pinging the host; Nagios will warn that you are not monitoring any services
- The filename can be anything ending **`.cfg`**
- Organise your devices however you like – e.g. related hosts in the same file

Generic host template

generic-host_nagios2.cfg

```
define host {
   name                            generic-host    ; The name of this host template
   notifications_enabled           1   ; Host notifications are enabled
   event_handler_enabled           1   ; Host event handler is enabled
   flap_detection_enabled          1   ; Flap detection is enabled
   failure_prediction_enabled      1   ; Failure prediction is enabled
   process_perf_data               1   ; Process performance data
   retain_status_information       1   ; Retain status information across program restarts
   retain_nonstatus_information    1   ; Retain non-status information across restarts
   check_command                   check-host-alive
   max_check_attempts              10
   notification_interval           0
   notification_period             24x7
   notification_options            d,u,r
   contact_groups                  admins
   register                        0   ; DON'T REGISTER THIS DEFINITION –
                                       ; IT'S NOT A REAL HOST, JUST A TEMPLATE!
}
```

Overriding defaults

All settings can be overridden per host

pcs.cfg

```
define host {
    host_name                   pc1
    alias                       pc1 in group 1
    address                     pc1.ws.nsrc.org
    use                         generic-host
    notification_interval       120
    contact_groups              admins,managers
}
```

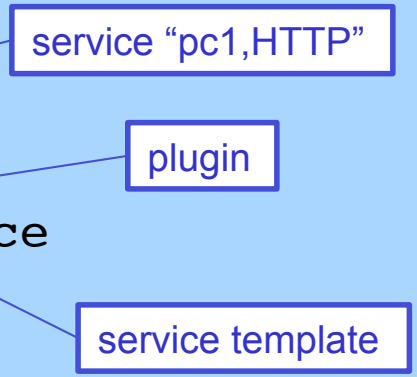
Defining services (direct way)

pcs.cfg

```
define host {
    host_name       pc1
    alias           pc1 in group 1
    address         pc1.ws.nsrc.org
    use             generic-host
}

define service {
    host_name               pc1
    service_description     HTTP
    check_command           check_http
    use                     generic-service
}

define service {
    host_name               pc1
    service_description     SSH
    check_command           check_ssh
    use                     generic-service
}
```



Service checks

- The combination of host + service is a unique identifier for the service check, e.g.
  - “pc1,HTTP”
  - “pc1,SSH”
  - “pc2,HTTP”
  - “pc2,SSH”
- *check_command* points to the plugin
- *service template* pulls in settings for how often the check is done, and who and when to alert

Generic service template

generic-service_nagios2.cfg

```
define service {
    name                            generic-service
    active_checks_enabled           1
    passive_checks_enabled          1
    parallelize_check               1
    obsess_over_service             0
    check_freshness                 1
    notifications_enabled           1
    event_handler_enabled           1
    flap_detection_enabled          1
    process_perf_data               1
    retain_status_information       1
    retain_nonstatus_information    1
    is_volatile                     0
    check_period                    24x7
    max_check_attempts              5
    normal_check_interval           5
    retry_check_interval            1
    notification_interval           60
    notification_period             24x7
    notification_options            w,u,c,r
    contact_groups                  admins
    register                        0
}
```

Overriding defaults

Again, settings can be overridden per service

services_nagios2.cfg

```
define service {
   host_name                pc1
   service_description      HTTP
   check_command            check_http
   use                      generic-service
   contact_groups           admins,managers
   max_check_attempts       3
}
```

Repeated service checks

- Often we are monitoring an identical service on many hosts
- To avoid duplication, a better way is to define a service check for all hosts in a *hostgroup*

Creating hostgroups

hostgroups_nagios2.cfg

```
define hostgroup {
   hostgroup_name   http-servers
   alias            HTTP servers
   members          pc1,pc2
}

define hostgroup {
   hostgroup_name   ssh-servers
   alias            SSH servers
   members          pc1,pc2
}
```

Monitoring services in hostgroups

services_nagios2.cfg

```
define service {
    hostgroup_name          http-servers
    service_description     HTTP
    check_command           check_http
    use                     generic-service
}

define service {
    hostgroup_name          ssh-servers
    service_description     SSH
    check_command           check_ssh
    use                     generic-service
}
```

e.g. if hostgroup “http-servers” contains pc1 and pc2 then Nagios creates HTTP service checks for both hosts. The service checks are called “pc1,HTTP” and “pc2,HTTP”

Alternative view

- Instead of saying “this hostgroup contains these PCs” you can say “this PC belongs to these hostgroups”
- No need for the “members” line in hostgroups file

Alternative group membership

pcs.cfg

```
define host {
    host_name     pc1
    alias         pc1 in group 1
    address       pc1.ws.nsrc.org
    use           generic-host
    hostgroups    ssh-servers,http-servers
}

define host {
    host_name     pc2
    alias         pc2 in group 1
    address       pc2.ws.nsrc.org
    use           generic-host
    hostgroups    ssh-servers,http-servers
}
```

Hosts and services conveniently defined in the same place

Other uses for hostgroups

Choosing icons for the status map

pcs.cfg

```
define host {
    host_name    pc1
    alias        pc1 in group 1
    address      pc1.ws.nsrc.org
    use          generic-host
    hostgroups   ssh-servers,http-servers,debian-servers
}
```

extinfo_nagios2.cfg

```
define hostextinfo {
    hostgroup_name    debian-servers
    notes             Debian GNU/Linux servers
    icon_image        base/debian.png
    statusmap_image   base/debian.gd2
}
```

Optional: servicegroups

- You can can also group together services into a “servicegroup”
- This is so related or dependent services can be viewed together in the web interface
- The services themselves must already exist

servicegroups.cfg

```
define servicegroup {
   servicegroup_name   mail-services
   alias         Services comprising the mail platform
   members       web1,HTTP,web2,HTTP,mail1,IMAP,db1,MYSQL
}
```

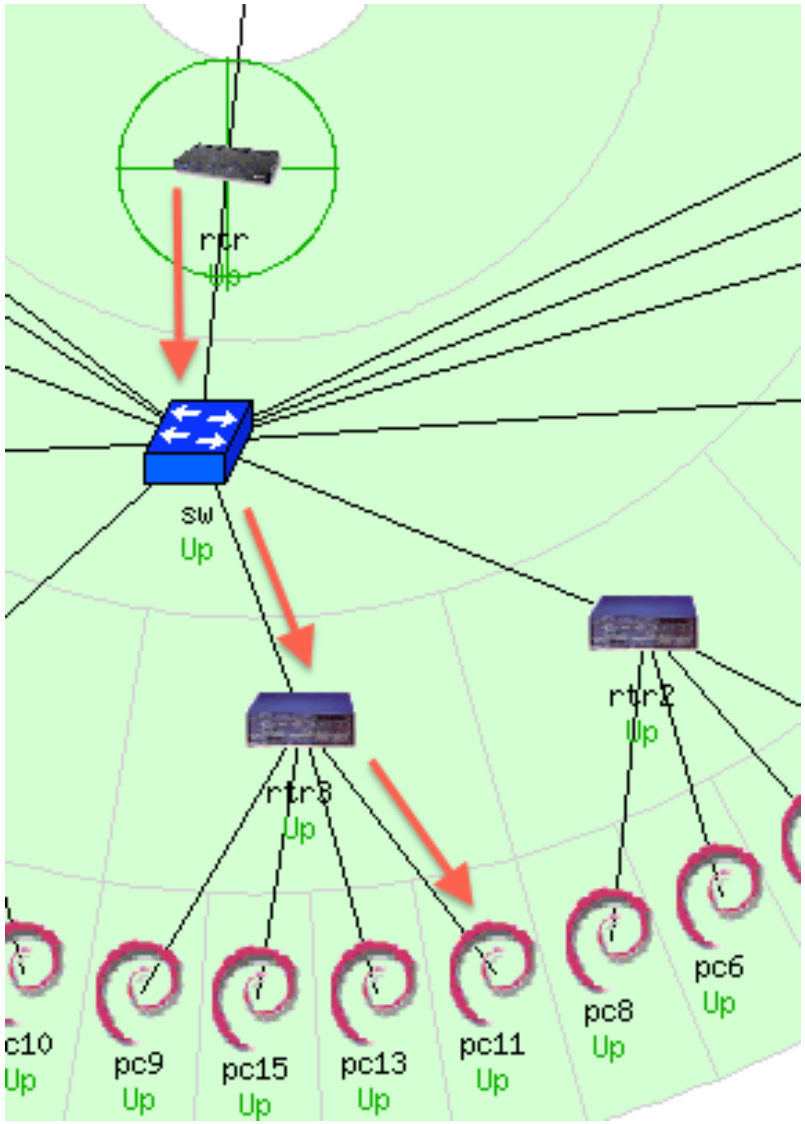
Configuring topology

pcs.cfg

```
define host {
    host_name    pc1
    alias        pc1 in group 1
    address      pc1.ws.nsrc.org
    use          generic-host
    parents      rtr1
}
```

- This means “pc1 is on the far side of rtr1”
- If rtr1 goes down, pc1 is marked “unreachable” rather than “down”
- Prevents a cascade of alerts if rtr1 goes down
- Also allows Nagios to draw cool status map

Another view of configuration



RTR

```
define host {
    use             generic-host
    host_name       rtr
    alias           Gateway Router
    address         10.10.0.254     }
```

SW

```
define host {
    use             generic-host
    host_name       sw
    alias           Backbone Switch
    address         10.10.0.253
    parents         rtr     }
```

RTR3

```
define host {
    use             generic-host
    host_name       rtr3
    alias           router 3
    address         10.10.3.254
    parents         sw      }
```

PC11…

Out-of-Band (OOB) notifications

A critical item to remember: an SMS or message system that is independent from your network.

- You can utilize a cell phone connected to the Nagios server, or a USB dongle with SIM card
- You can use packages like:

gammu: http://wammu.eu/
gnokii: http://www.gnokii.org/
sms-tools: http://smstools3.kekekasvi.com/

References

- **Nagios web site**
  http://www.nagios.org/
- **Nagios plugins site**
  http://www.nagiosplugins.org/
- *Nagios System and Network Monitoring*, by Wolfgang Barth. Good book about Nagios.
- **Unofficial Nagios plugin site**
  http://nagios.exchange.org/
- **A Debian tutorial on Nagios**
  http://www.debianhelp.co.uk/nagios.htm
- **Commercial Nagios support**
  http://www.nagios.com/

Questions?

?

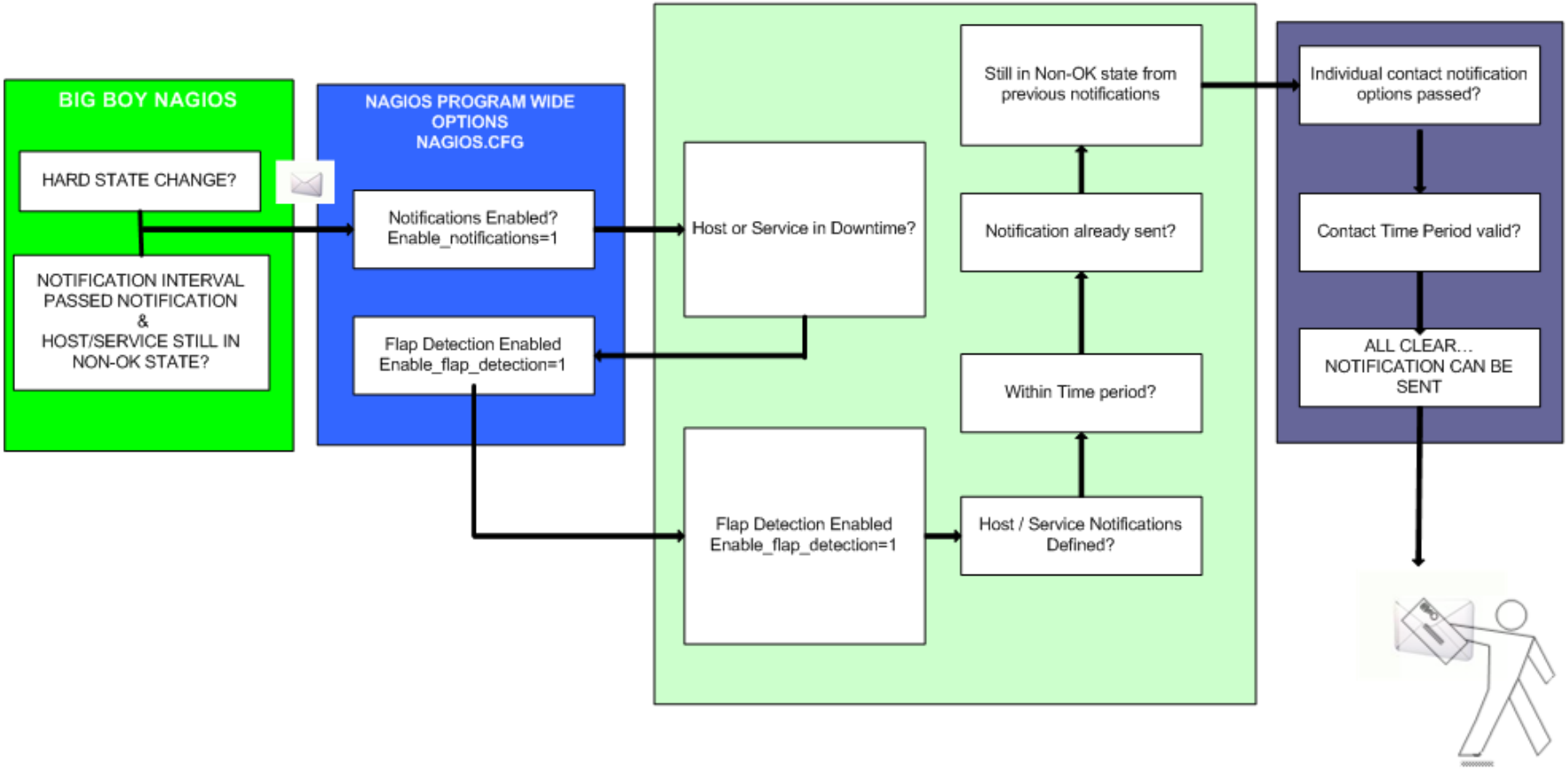
Additional Details

A few additional slides you may find useful or informative…

Features, features, features…

- Allows you to acknowledge an event.
  - A user can add comments via the GUI
- You can define maintenance periods
  - By device or a group of devices
- Maintains availability statistics and generates reports
- Can detect flapping and suppress additional notifications.
- Allows for multiple notification methods:
  - e-mail, pager, SMS, winpopup, audio, etc...
- Allows you to define notification levels for escalation

NAGIOS - NOTIFICATION FLOW DIAGRAM



NOTE: The flow will only continue when each of the listed filters are satisfied.

CONTACT GETS THE NOTIFICATION MESSAGE

Notification Options (Host)

Host state:

When configuring a host you can be notified on the following conditions:

- **d:** DOWN
- **u:** UNREACHABLE
- **r:** RECOVERY
- **f:** FLAPPING (start/end)
- **s:** SCHEDULED DOWNTIME (start/end)
- **n:** NONE

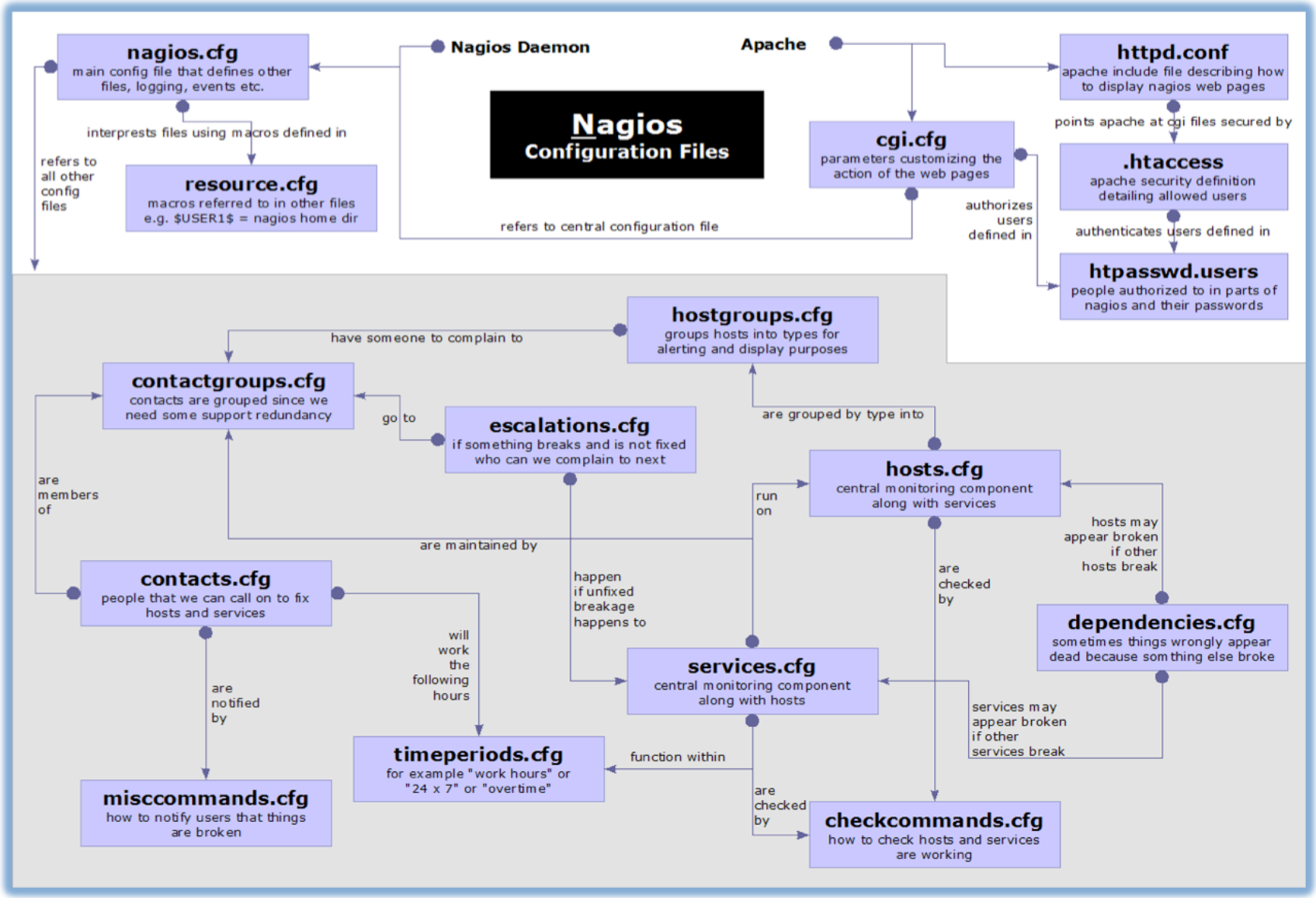
Notification Options (Service)

Service state:

When configuring a service you can be notified on the following conditions:

- **w:** WARNING
- **c:** CRITICAL
- **u:** UNKNOWN
- **r:** RECOVERY
- **f:** FLAPPING (start/end)
- **s:** SCHEDULED DOWNTIME (start/end)
- **n:** NONE

Configuration files (Official)



Debian/Ubuntu config file layout

Located in /etc/nagios3/

Important files include:

- nagios.cfg — Main configuration file.
- cgi.cfg — Controls the web interface and security options.
- commands.cfg — The commands that Nagios uses for notifications.
- conf.d/* — All other configuration goes here!

Configuration files continued

Under conf.d/*

contacts_nagios2.cfg	users and groups
extinfo_nagios2.cfg	make your UI pretty
generic-host_nagios2.cfg	default host template
generic-service_nagios2.cfg	default service template
host-gateway_nagios3.cfg	upstream router definition
hostgroups_nagios2.cfg	groups of nodes
localhost_nagios2.cfg	definition of nagios host
services_nagios2.cfg	what services to check
timeperiods_nagios2.cfg	when to check who to notify

Configuration files continued

Under conf.d some other possible config files:

- servicegroups.cfg — Groups of nodes and services
- pcs.cfg — Sample definition of PCs (hosts)
- switches.cfg — Definitions of switches (hosts)
- routers.cfg — Definitions of routers (hosts)

Main configuration details

Global settings

File: `/etc/nagios3/nagios.cfg`

- Says where other configuration files are.
- General Nagios behavior:
  - For large installations you should tune the installation via this file.
  - See: *Tunning Nagios for Maximum Performance* **http://nagios.sourceforge.net/docs/3_0/tuning.html**

CGI configuration

/etc/nagios3/cgi.cfg

- You can change the CGI directory if you wish
- Authentication and authorization for Nagios use:
  - Activate authentication via Apache's .htpasswd mechanism, or using RADIUS or LDAP.
  - Users can be assigned rights via the following variables:
    - authorized_for_system_information
    - authorized_for_configuration_information
    - authorized_for_system_commands
    - authorized_for_all_services
    - authorized_for_all_hosts
    - authorized_for_all_service_commands
    - authorized_for_all_host_commands

Time Periods

This defines the base periods that control checks, notifications, etc.

- Defaults: 24 x 7
- Could adjust as needed, such as work-week only.
- Could adjust a new time period for “outside of regular hours”, etc.

```
# '24x7'
define timeperiod{
        timeperiod_name 24x7
        alias           24 Hours A Day, 7 Days A Week
        sunday          00:00-24:00
        monday          00:00-24:00
        tuesday         00:00-24:00
        wednesday       00:00-24:00
        thursday        00:00-24:00
        friday          00:00-24:00
        saturday        00:00-24:00
        }
```

Configuring service/host checks

/etc/nagios-plugins/config/ssh.cfg

```
define command {
  command_name  check_ssh
  command_line  /usr/lib/nagios/plugins/check_ssh '$HOSTADDRESS$'
}

define command {
  command_name  check_ssh_port
  command_line  /usr/lib/nagios/plugins/check_ssh -p '$ARG1$' '$HOSTADDRESS$'
}
```

- Notice the same plugin can be invoked in different ways (“commands”)
- Command and arguments are separated by exclamation marks (!)
- e.g. to check SSH on a non-standard port, you can do it like this:

```
define service {
   hostgroup_name          ssh-servers-2222
   service_description     SSH-2222
   check_command           check_ssh_port!2222
   use                     generic-service
}
```

this is $ARG1$

Notification commands

Allows you to utilize any command you wish.
We could use this to generate tickets in RT.

```
# 'notify-by-email' command definition
define command{
        command_name    notify-by-email
        command_line    /usr/bin/printf "%b" "Service: $SERVICEDESC$\nHost:
$HOSTNAME$\nIn: $HOSTALIAS$\nAddress: $HOSTADDRESS$\nState: $SERVICESTATE$
\nInfo: $SERVICEOUTPUT$\nDate: $SHORTDATETIME$" | /bin/mail -s
'$NOTIFICATIONTYPE$: $HOSTNAME$/$SERVICEDESC$ is $SERVICESTATE$'
$CONTACTEMAIL$
        }
```

From: nagios@nms.localdomain
To: router_group@localdomain
Subject: Host DOWN alert for TLD1-RTR!
Date: Thu, 29 Jun 2006 15:13:30 -0700

Host: gw-rtr
In: Core_Routers
State: DOWN
Address: 192.0.2.100
Date/Time: 06-29-2006 15:13:30
Info: CRITICAL - Plugin timed out after 6 seconds

Group service configuration

```
# check that ssh services are running
define service {
        hostgroup_name              ssh-servers
        service_description         SSH
        check_command               check_ssh
        use                         generic-service
        notification_interval       0
}
```

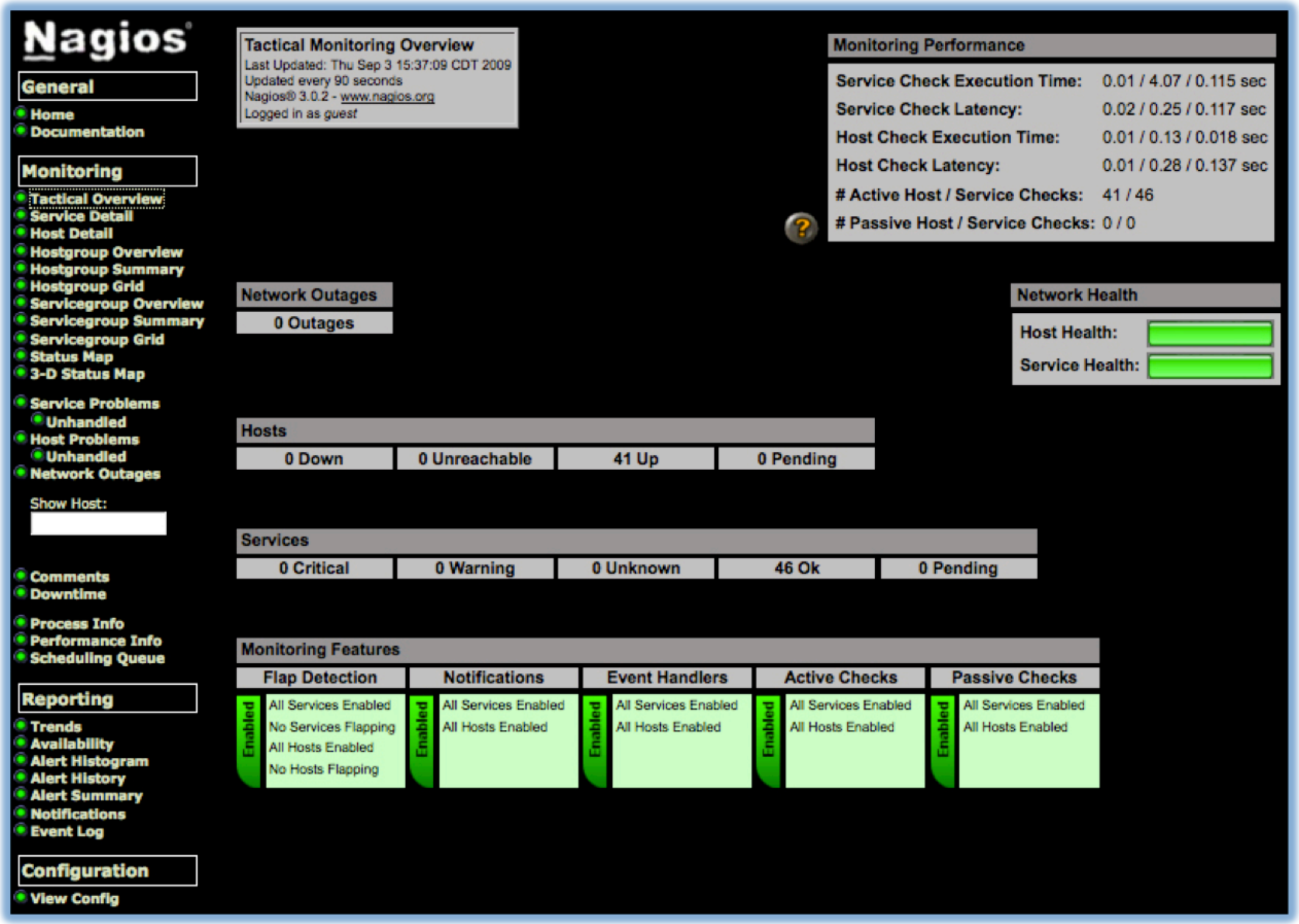
The “service_description” is important if you plan to create Service Groups. Here is a sample Service Group definition:

```
define servicegroup{
   servicegroup_name    Webmail
   alias                web-mta-storage-auth
   members              srvr1,HTTP,srvr1,SMTP,srvr1,POP,   \
                        srvr1,IMAP,srvr1,RAID,srvr1,LDAP,  \
                        srvr2,HTTP,srvr2,SMTP,srvr2,POP,   \
                        srvr2,IMAP,srvr2,RAID,srvr2,LDAP
}
```

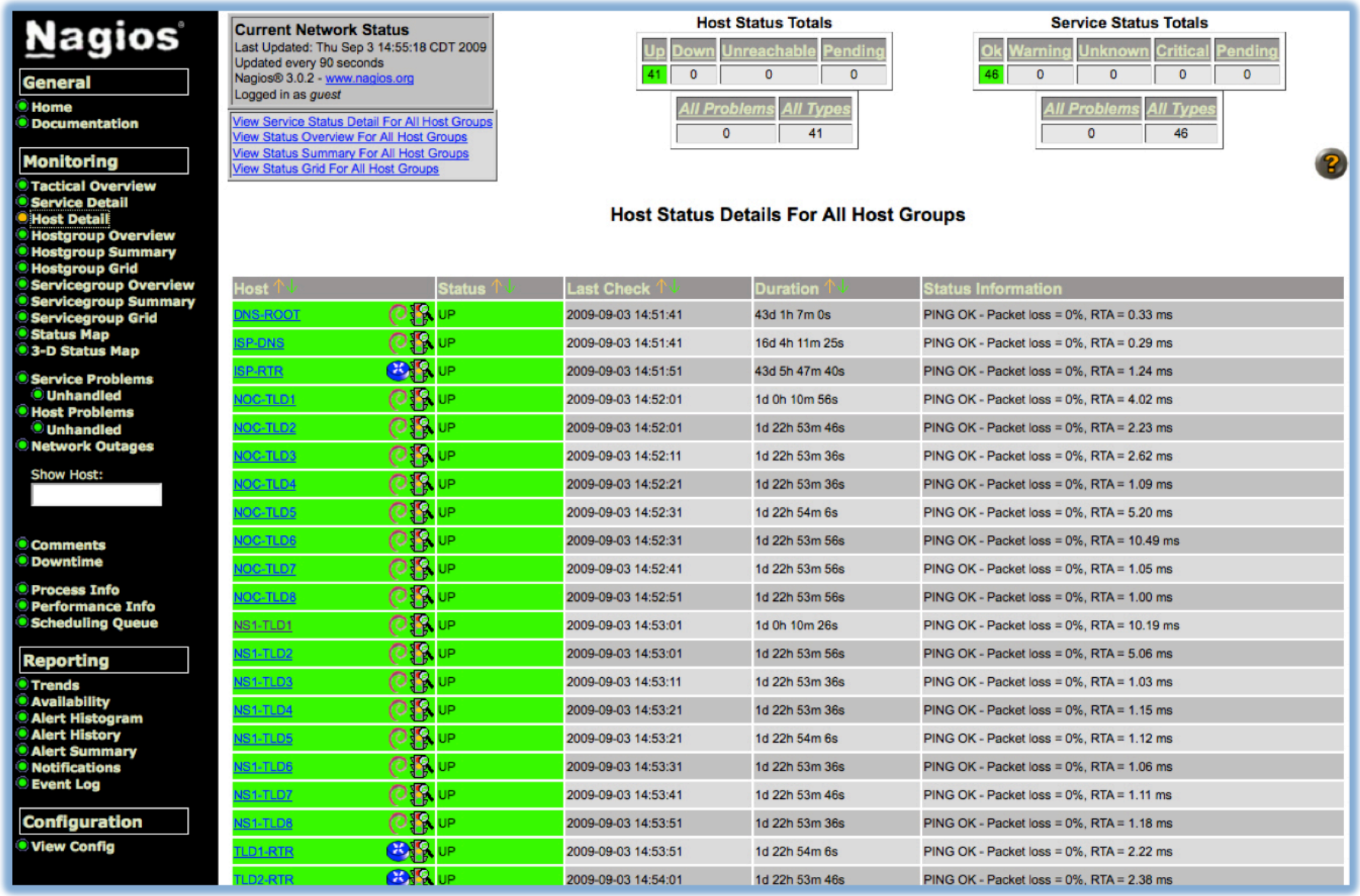
Screen Shots

A few sample screen shots from a Nagios install.

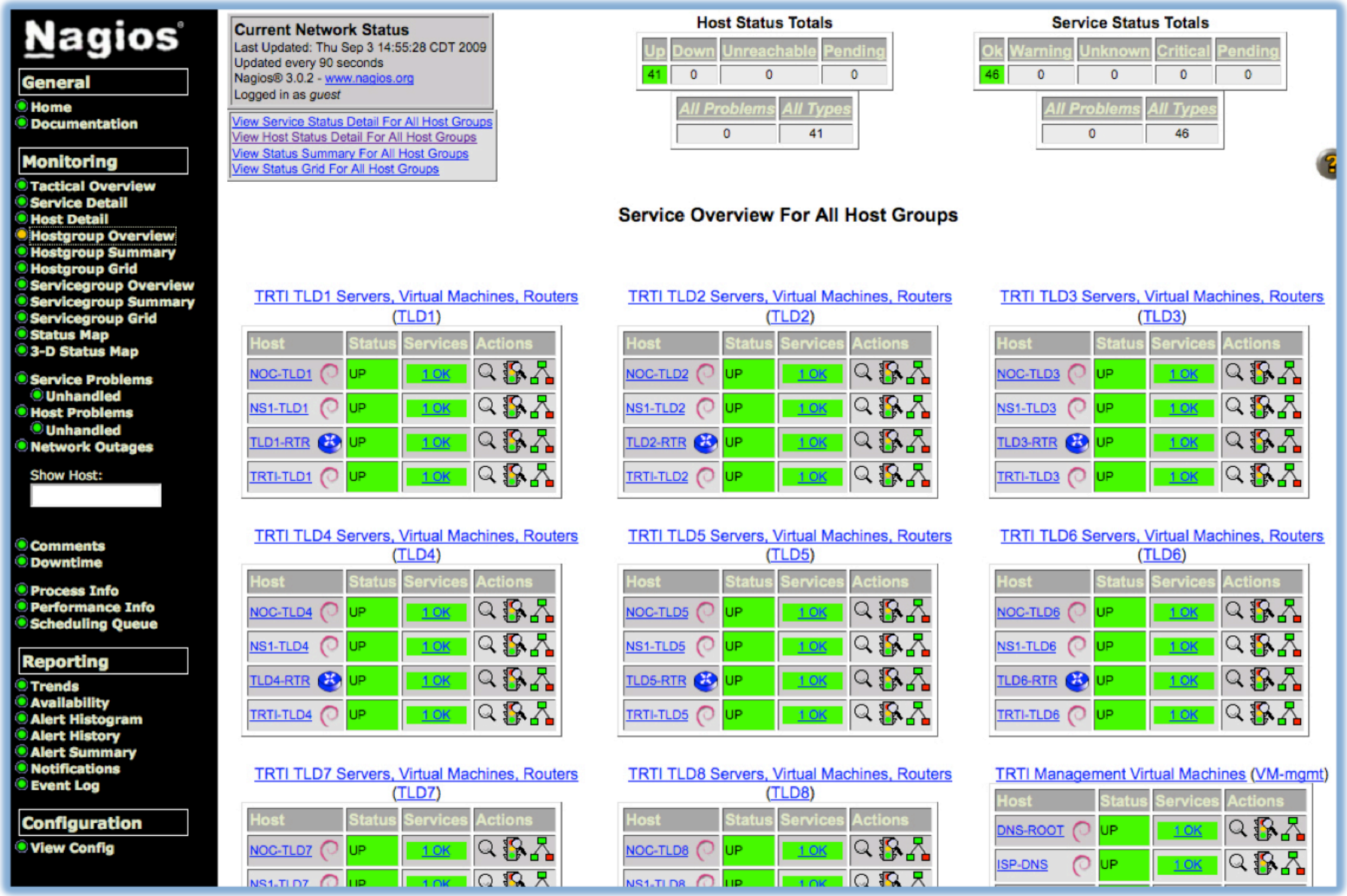
General View



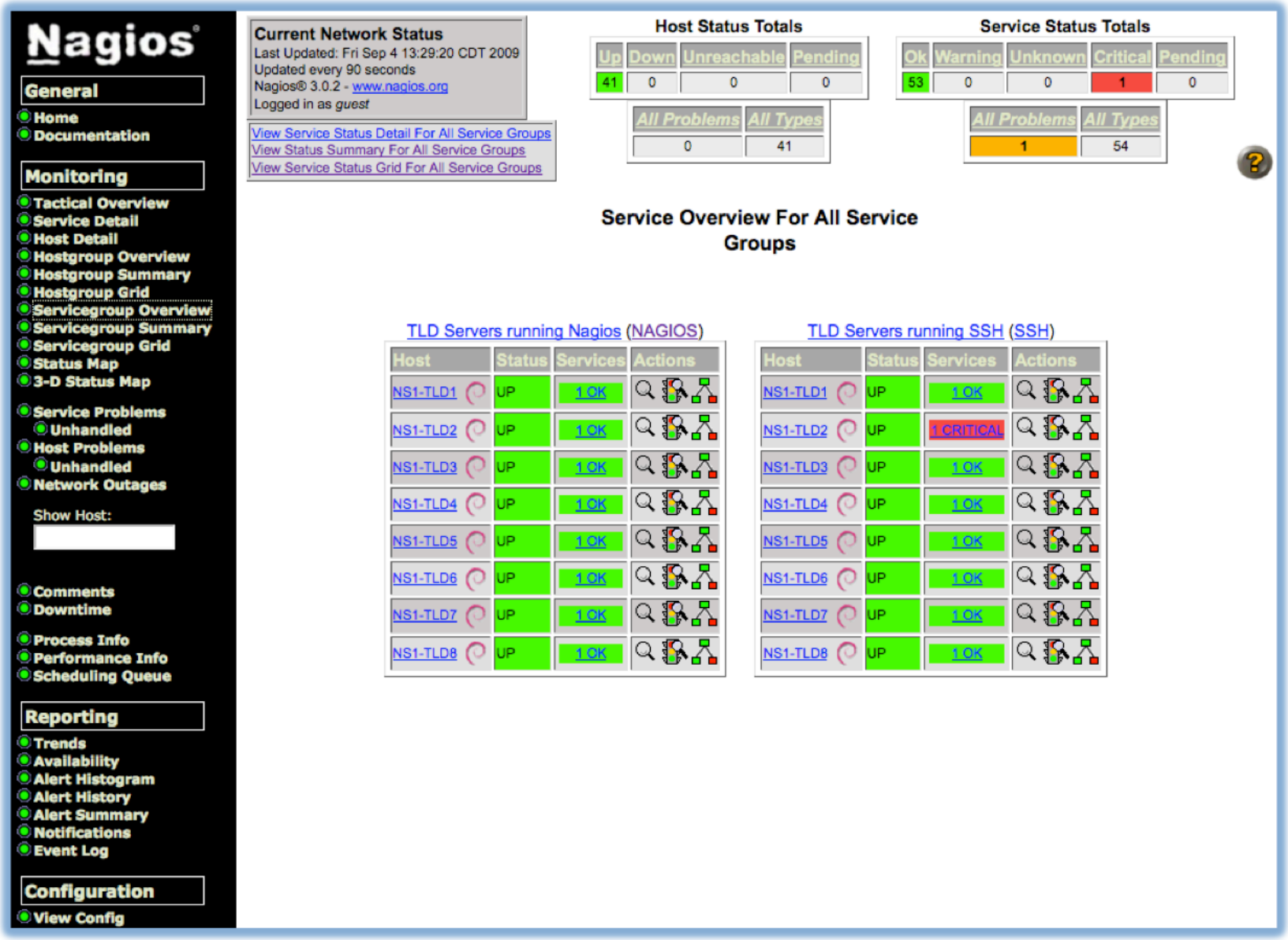
Host Detail


Nagios
General
Home
Documentation
Monitoring
Tactical Overview
Service Detail
Host Detail
Hostgroup Overview
Hostgroup Summary
Hostgroup Grid
Servicegroup Overview
Servicegroup Summary
Servicegroup Grid
Status Map
3-D Status Map
Service Problems
Unhandled
Host Problems
Unhandled
Network Outages
Show Host:
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Reporting
Trends
Availability
Alert Histogram
Alert History
Alert Summary
Notifications
Event Log
Configuration
View Config
Current Network Status
Last Updated: Thu Sep 3 14:55:18 CDT 2009
Updated every 90 seconds
Nagios® 3.0.2 - www.nagios.org
Logged in as guest
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 Totals
Up Down Unreachable Pending
41 0 0 0
All Problems All Types
0 41
Service Status Totals
Ok Warning Unknown Critical Pending
46 0 0 0 0
All Problems All Types
0 46
Host Status Details For All Host Groups
Host Status Last Check Duration Status Information
DNS-ROOT UP 2009-09-03 14:51:41 43d 1h 7m 0s PING OK - Packet loss = 0%, RTA = 0.33 ms
ISP-DNS UP 2009-09-03 14:51:41 16d 4h 11m 25s PING OK - Packet loss = 0%, RTA = 0.29 ms
ISP-RTR UP 2009-09-03 14:51:51 43d 5h 47m 40s PING OK - Packet loss = 0%, RTA = 1.24 ms
NOC-TLD1 UP 2009-09-03 14:52:01 1d 0h 10m 56s PING OK - Packet loss = 0%, RTA = 4.02 ms
NOC-TLD2 UP 2009-09-03 14:52:01 1d 22h 53m 46s PING OK - Packet loss = 0%, RTA = 2.23 ms
NOC-TLD3 UP 2009-09-03 14:52:11 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 2.62 ms
NOC-TLD4 UP 2009-09-03 14:52:21 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 1.09 ms
NOC-TLD5 UP 2009-09-03 14:52:31 1d 22h 54m 6s PING OK - Packet loss = 0%, RTA = 5.20 ms
NOC-TLD6 UP 2009-09-03 14:52:31 1d 22h 53m 56s PING OK - Packet loss = 0%, RTA = 10.49 ms
NOC-TLD7 UP 2009-09-03 14:52:41 1d 22h 53m 56s PING OK - Packet loss = 0%, RTA = 1.05 ms
NOC-TLD8 UP 2009-09-03 14:52:51 1d 22h 53m 56s PING OK - Packet loss = 0%, RTA = 1.00 ms
NS1-TLD1 UP 2009-09-03 14:53:01 1d 0h 10m 26s PING OK - Packet loss = 0%, RTA = 10.19 ms
NS1-TLD2 UP 2009-09-03 14:53:01 1d 22h 53m 56s PING OK - Packet loss = 0%, RTA = 5.06 ms
NS1-TLD3 UP 2009-09-03 14:53:11 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 1.03 ms
NS1-TLD4 UP 2009-09-03 14:53:21 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 1.15 ms
NS1-TLD5 UP 2009-09-03 14:53:21 1d 22h 54m 6s PING OK - Packet loss = 0%, RTA = 1.12 ms
NS1-TLD6 UP 2009-09-03 14:53:31 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 1.06 ms
NS1-TLD7 UP 2009-09-03 14:53:41 1d 22h 53m 46s PING OK - Packet loss = 0%, RTA = 1.11 ms
NS1-TLD8 UP 2009-09-03 14:53:51 1d 22h 53m 36s PING OK - Packet loss = 0%, RTA = 1.18 ms
TLD1-RTR UP 2009-09-03 14:53:51 1d 22h 54m 6s PING OK - Packet loss = 0%, RTA = 2.22 ms
TLD2-RTR UP 2009-09-03 14:54:01 1d 22h 53m 46s PING OK - Packet loss = 0%, RTA = 2.38 ms

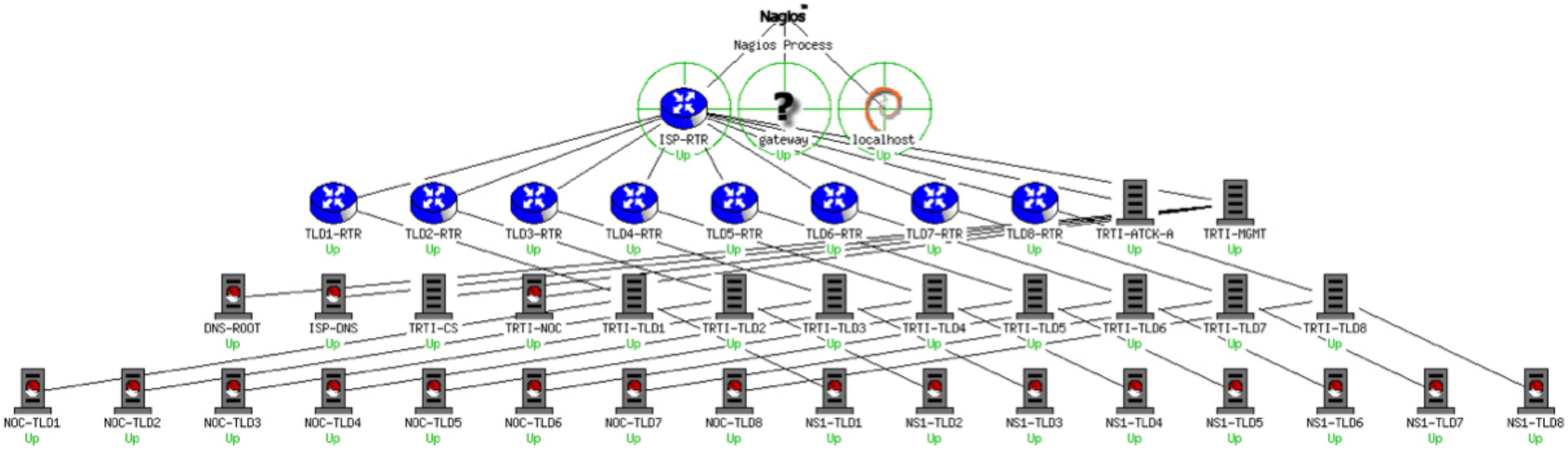
Host Groups Overview



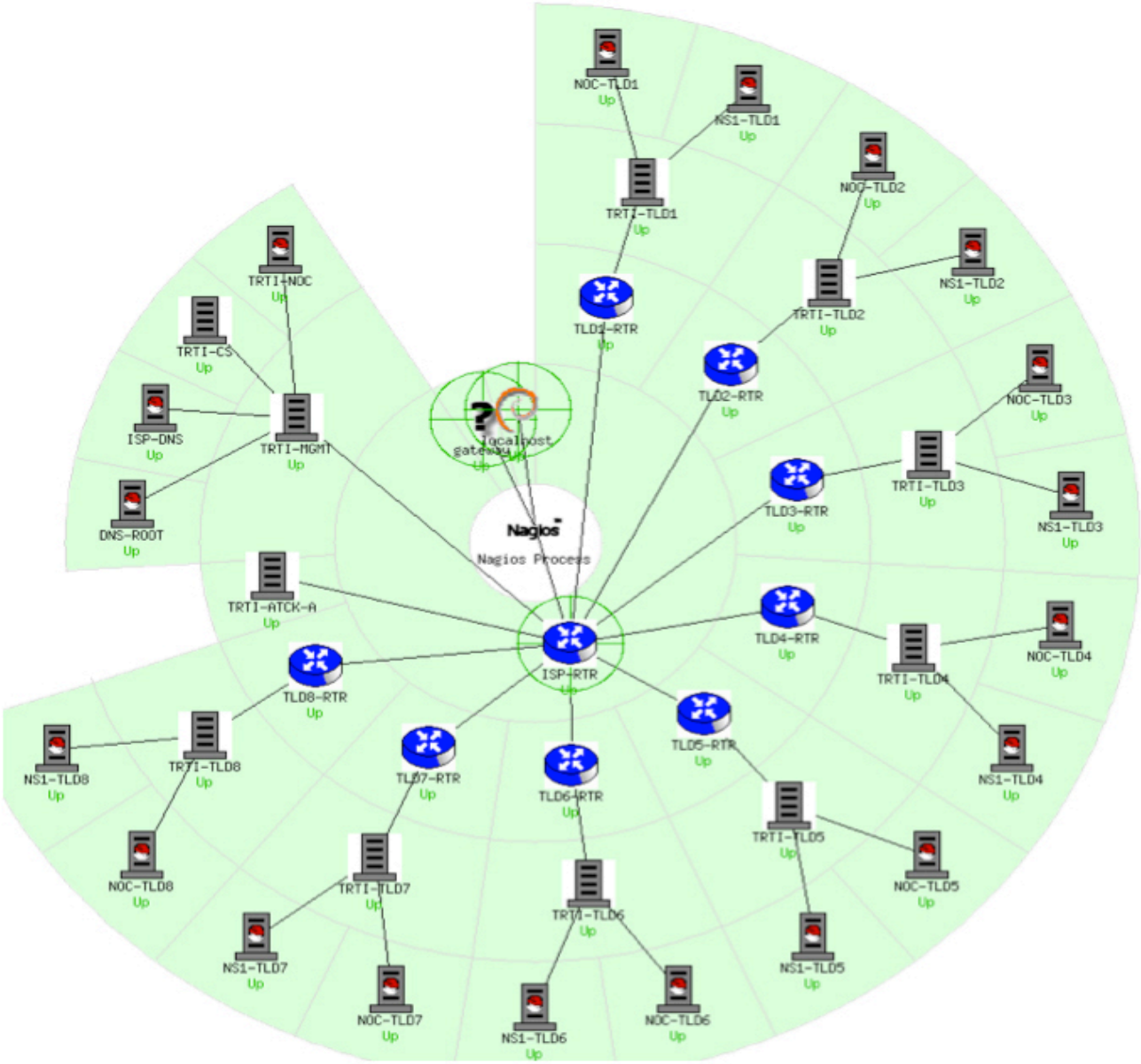
Service Groups Overview



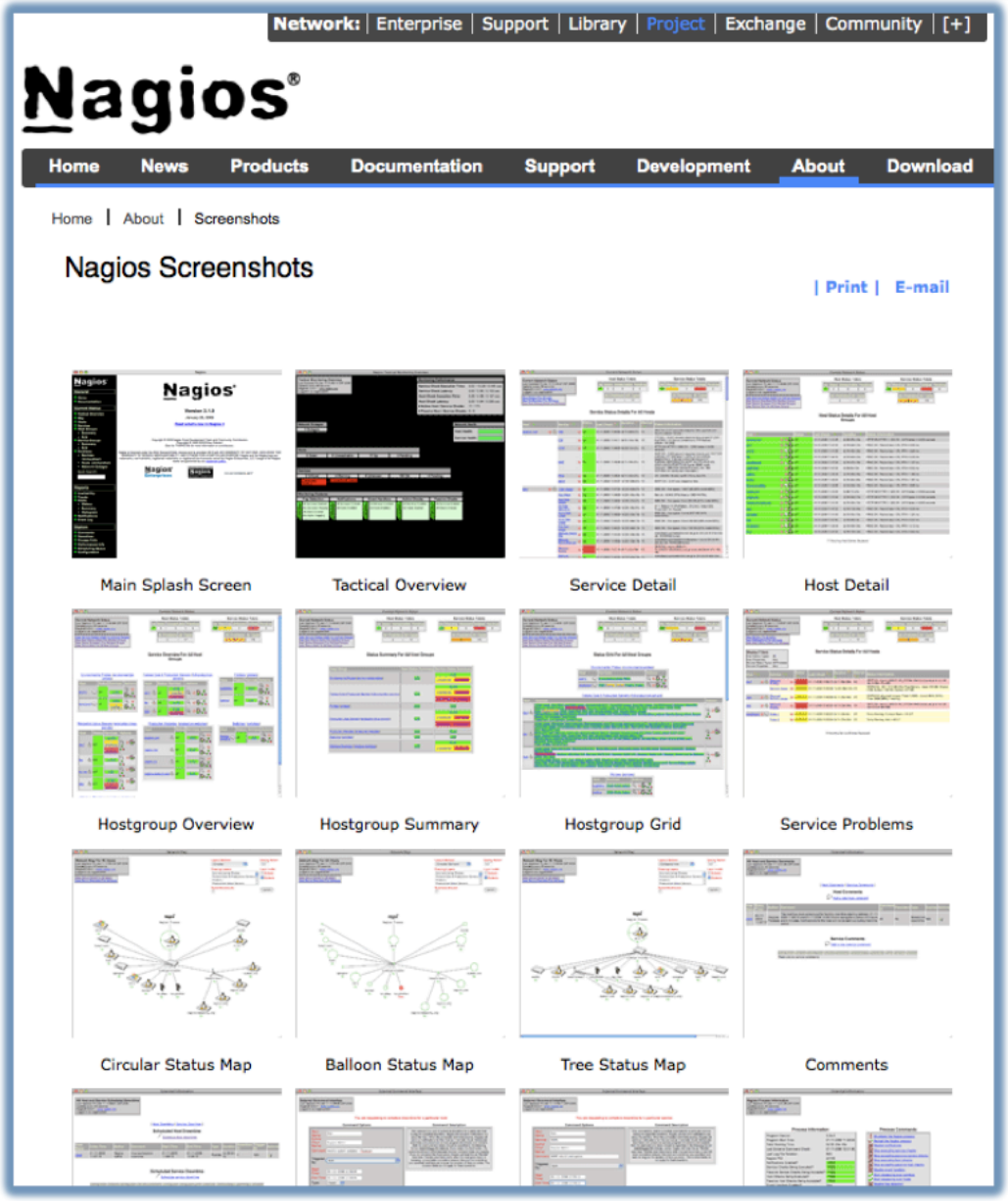
Collapsed tree status map



Marked-up circular status map



More sample screenshots



Many more sample Nagios screenshots available here:

http://www.nagios.org/about/screenshots