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

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

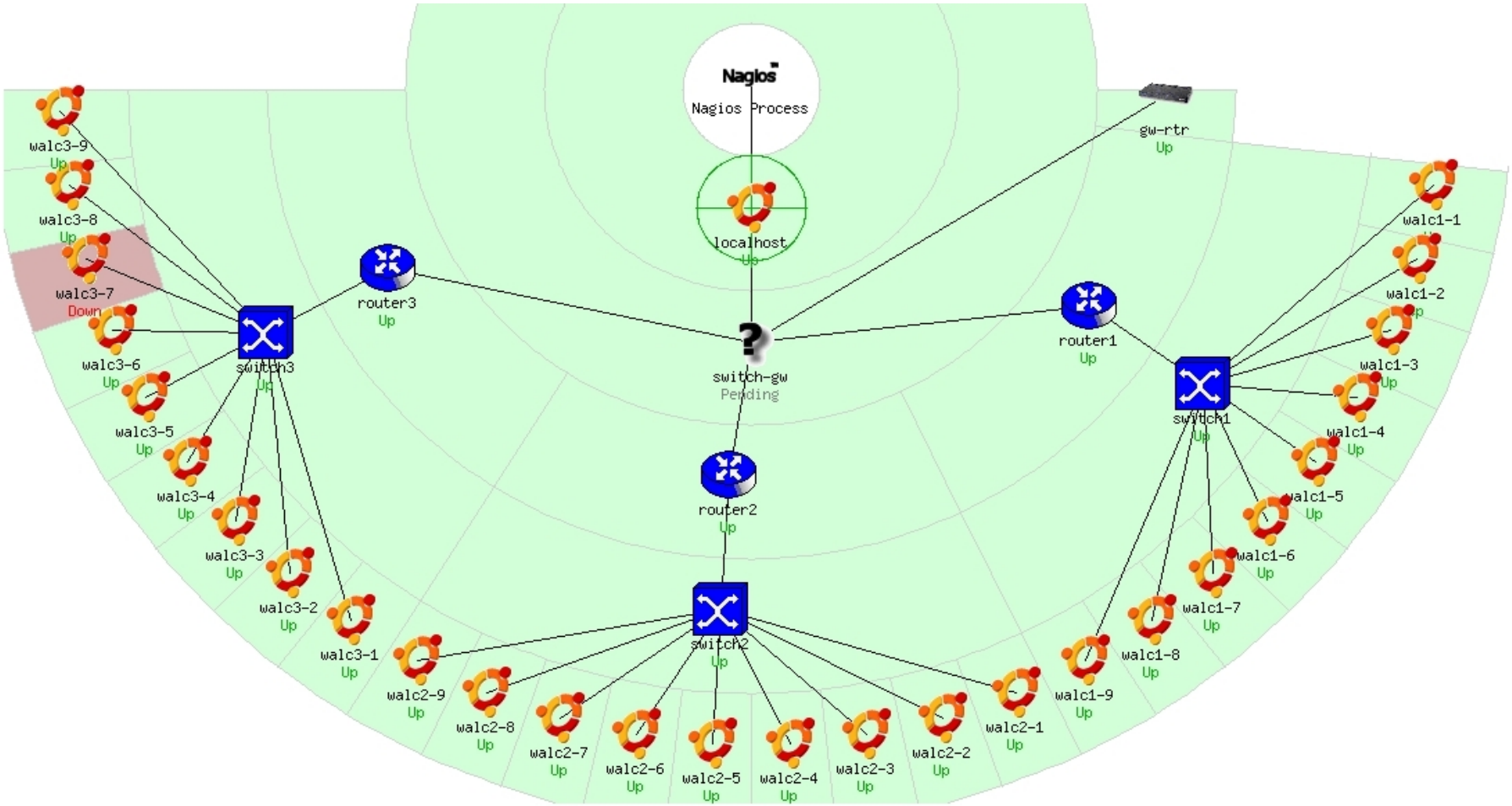
Hosts can have parents:

- 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

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
}
```

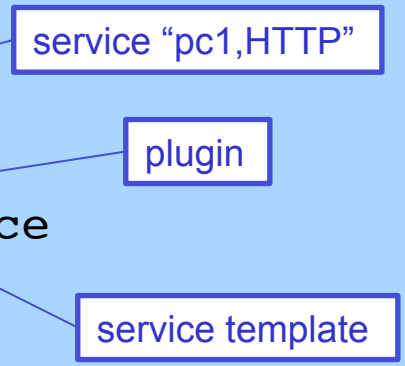
Defining services (direct way)

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             1
        check_freshness                 0
        notifications_enabled           1
        event_handler_enabled           1
        flap_detection_enabled          1
        failure_prediction_enabled      1
        process_perf_data               1
        retain_status_information       1
        retain_nonstatus_information    1
          notification_interval           0
          is_volatile                     0
          check_period                    24x7
          normal_check_interval           5
          retry_check_interval            1
          max_check_attempts              4
          notification_period             24x7
          notification_options            w,u,c,r
          contact_groups                  admins
        register                        0    ; DONT REGISTER THIS DEFINITION
        }
```

*Comments have been removed.

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”

Configuring topology

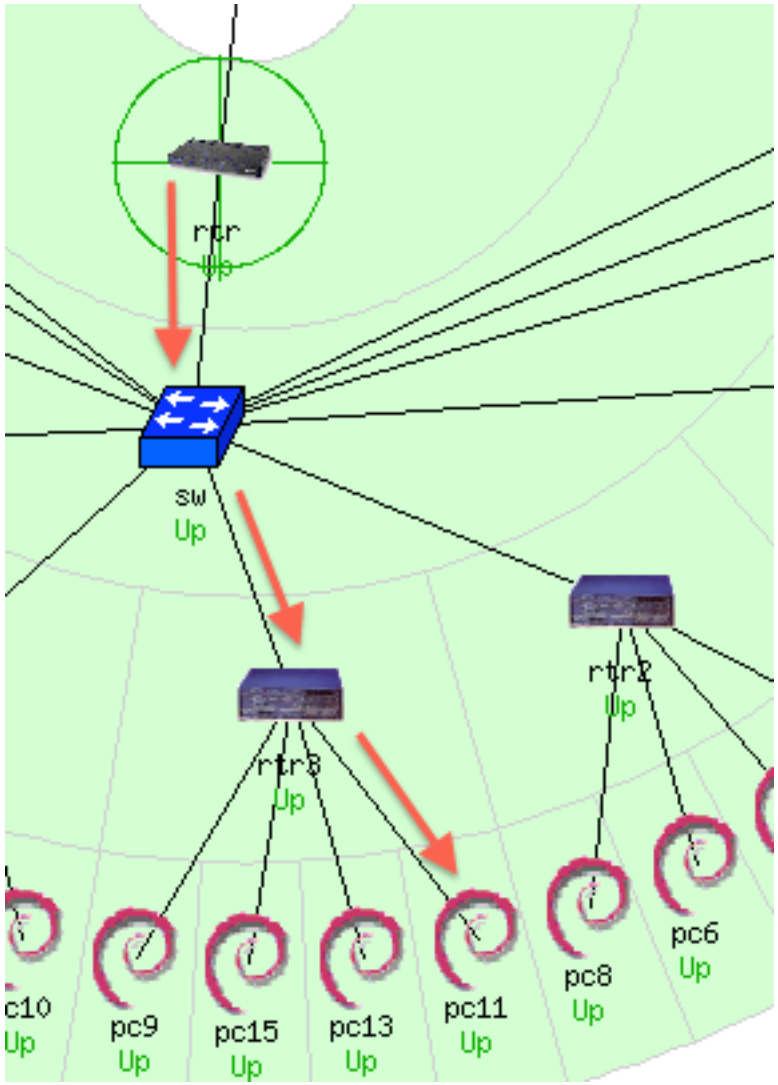
pcs.cfg

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

parent host

- 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…

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?

?