APNIC

RRDtool Fundamentals

Further information | 00 Example 2020

Trainer Name

Table of Content

RRDtool - Round Robin Database tool

- RRDtool
  - is a powerful tool to store time series data and create graphs
  - correlates time-series data like network bandwidth, temperatures, CPU load or any other data type
  - is used in many monitoring solutions and one of the easiest and best ways to store datapoints and generate graphs from them
- The back end of many popular graphing programs like Cacti, SmokePing, MRTG, Nagios, LibreNMS and others are based on RRDtool
- We use RRDtool to store and process data collected via SNMP

RRDtool - Round Robin Database tool

- RRDtool does 3 things
  - Creating Round-Robin Databases (RRDs)
  - Adding data to them
  - And creating graphs based on data in those databases
- Install RRDtool in Ubuntu

```
$ sudo apt install rrdtool -y
```

RRDtool - Create Graph

- Creating Graph manually

1. Creating an rrd database file

```
$ rrdtool create datafile.rrd \
        DS:packets:ABSOLUTE:900:0:10000000 \
        RRA:AVERAGE:0.5:1:9600 \
        RRA:AVERAGE:0.5:4:9600 \
        RRA:AVERAGE:0.5:24:6000
```

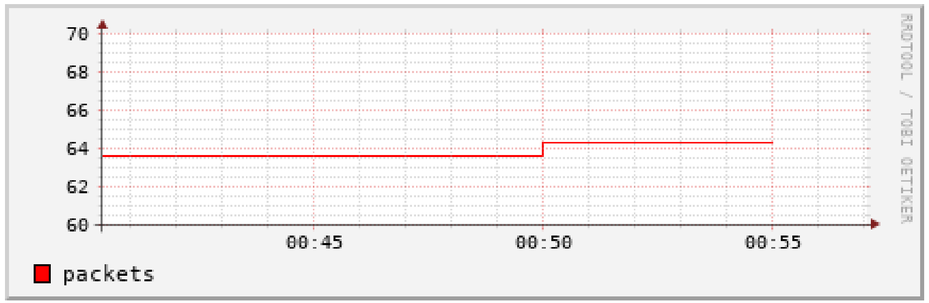
2. Fill the database with example data

```
$ START=$(expr $(date "+%s") - 1000)
COUNT=1000
for (( i = 0; i < ${COUNT}; i++ )); do
    VALUE=$(echo "scale=3; s($i/10) * 100" |
bc -l)
    rrdtool update datafile.rrd
${START}:${VALUE}
    START=$(expr ${START} + 1)
done
```

3. Create graph

```
$ rrdtool graph rrdtool1_graph1.png \
    --start now-1000s --end now \
    DEF:pkt=datafile.rrd:packets:AVERAGE \
    LINE1:pkt#FF0000:"packets"
```

4. Graph



https://oss.oetiker.ch/rrdtool/doc/rrdcreate.en.html

RRDtool - Create Graph

- rrdtool create datafile.rrd
  - Create a new database called datafile.rd
- DS:packets:ABSOLUTE:900:0:10000000
  - DS is a definition for a data source called “packets”
  - DS:ds-name:DST:heartbeat:min:max
    - DST means Data source type
- RRA:AVERAGE:0.5:1:9600
  - RRA means round robin archive
  - RRA:CF:xff:steps:rows
    - CF can be AVERAGE, MIN, MAX, LAST

- Creating Graph manually

1. Creating an rrd database file

```
$ rrdtool create datafile.rrd \
        DS:packets:ABSOLUTE:900:0:10000000 \
        RRA:AVERAGE:0.5:1:9600 \
        RRA:AVERAGE:0.5:4:9600 \
        RRA:AVERAGE:0.5:24:6000
```

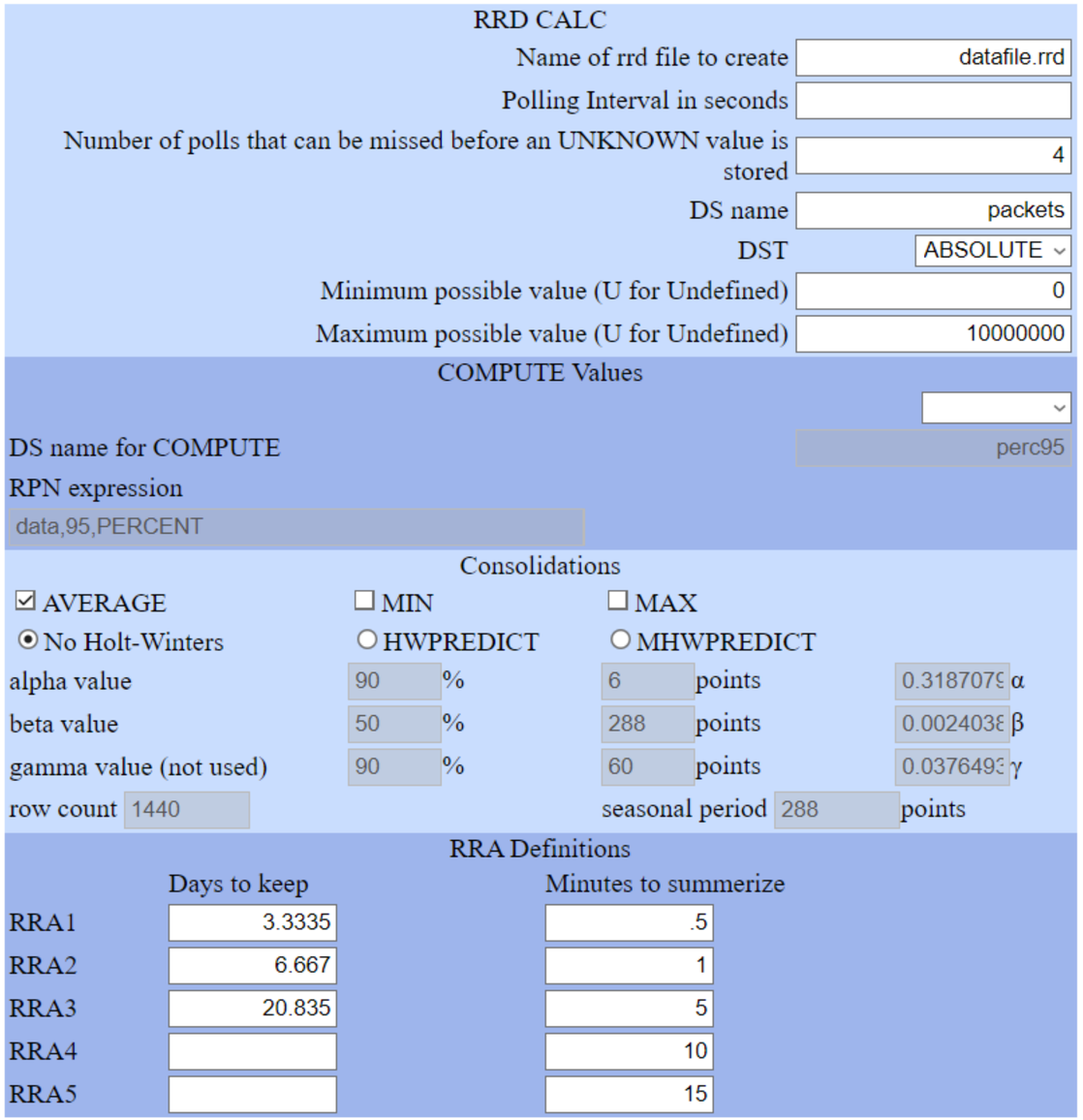
https://apfelboymchen.net/gnu/rrd/create/

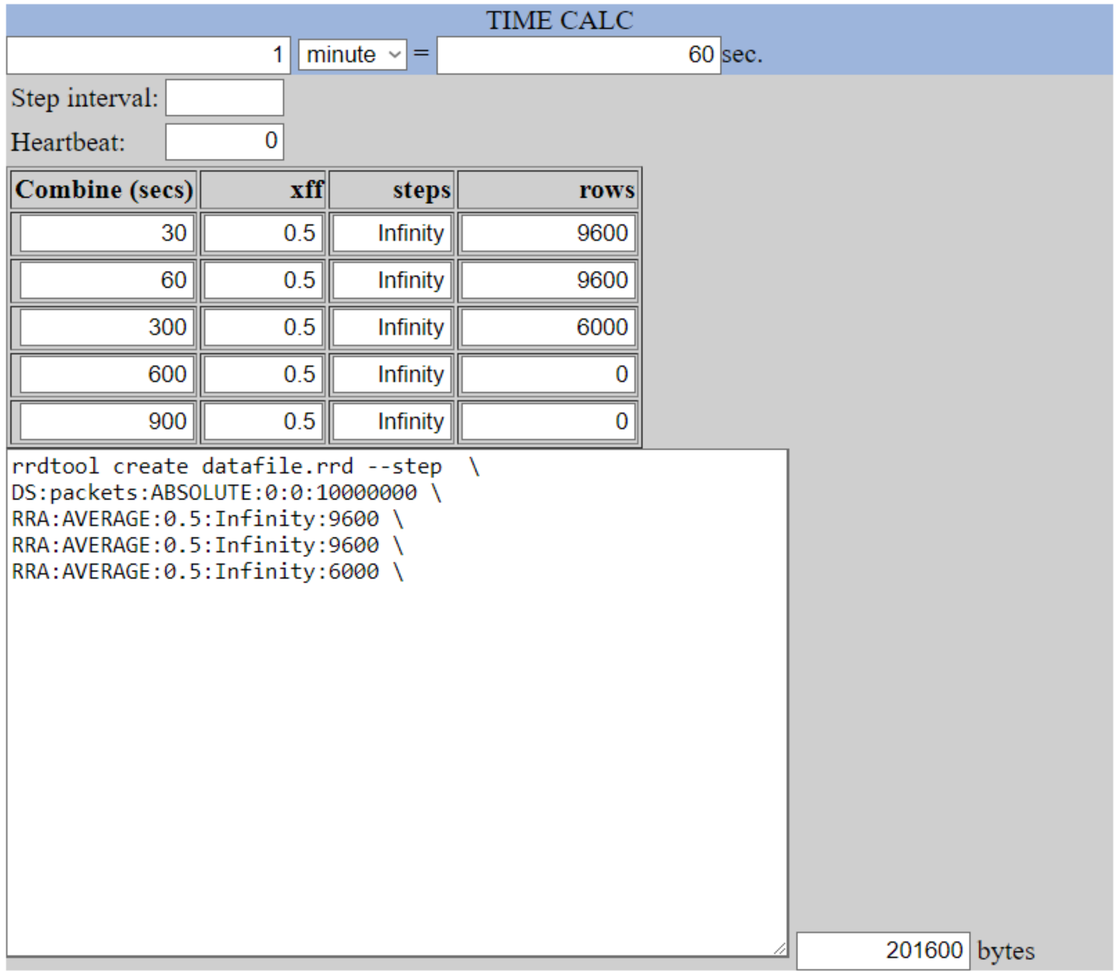
RRDtool - Create Graph

- There are 3 'Archives' inside this rrd
  - 1 measurement -> RRA:AVERAGE:0.5:1:9600
  - 4 measurements -> RRA:AVERAGE:0.5:4:9600
  - 24 measurement -> RRA:AVERAGE:0.5:24:6000
- How many values of that type that should be stored
  - RRA:AVERAGE:0.5:1:9600
    - 9600 values of 15-minute periods gives you 100 days
  - RRA:AVERAGE:0.5:4:9600
    - 9600 values of 1 hour (4 x 15mins) gives you 400 days
  - RRA:AVERAGE:0.5:24:6000
    - 6000 values of 6 hour-averages (24 x 15mins) gives you 1500 days

https://berthub.eu/rrd-mini-howto/cvs/rrd-mini-howto/output/rrd-mini-howto-1.html

RRDtool - Create Graph





https://eccentric.one/misc/rrdcalc.html

RRDtool – NMM Tools

- There are applications which leverages the RRDtool
- Will discuss two most common NMM tools for monitoring latency & uptime
  - SmokePing
  - Nagios

RRDtool Fundamentals

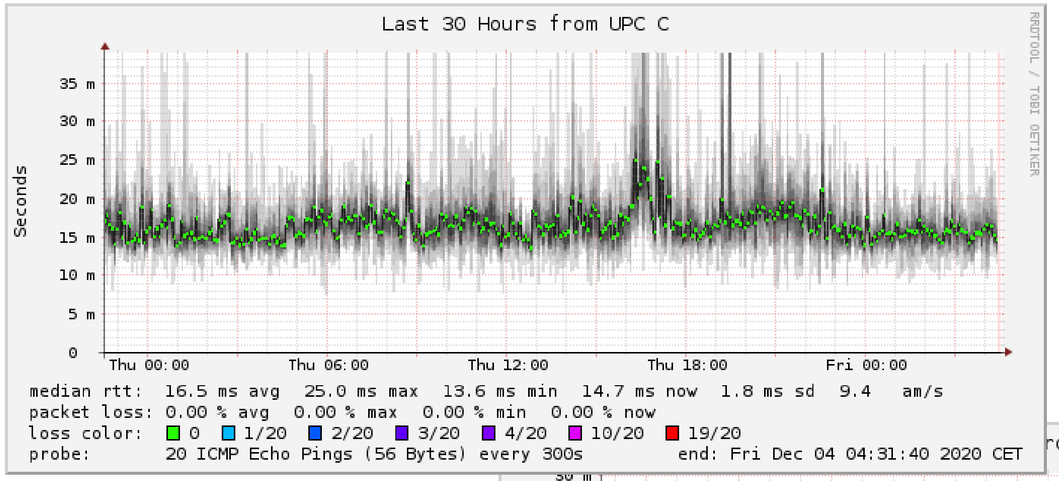
Tool: SmokePing

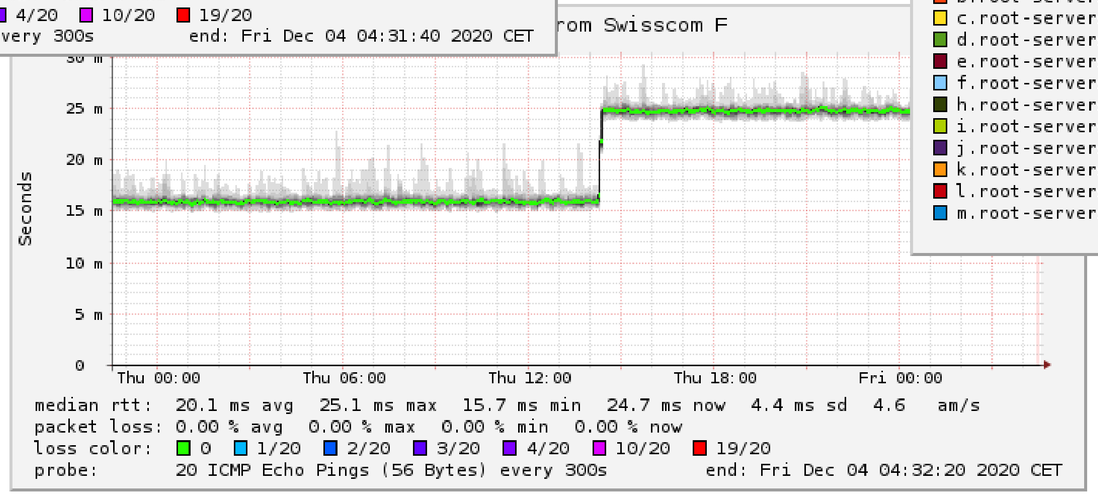
SmokePing

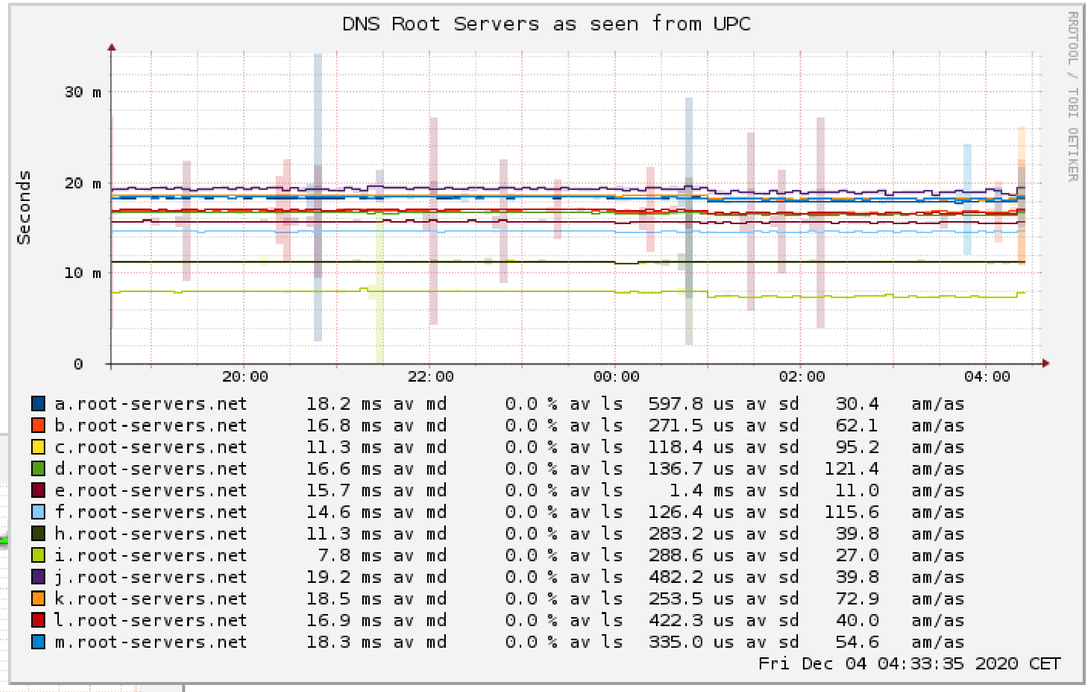
- SmokePing keeps track of network latency:
  - Excellent latency visualisation
  - Interactive graph explorer
  - Wide range of latency measurement plugins
  - Master/Slave System for distributed measurement
  - Highly configurable alerting system
  - Live Latency Charts with the most 'interesting' graphs
  - Free and OpenSource Software written in Perl written by Tobi Oetiker, the creator of MRTG and RRDtool

SmokePing - Graphs

- Demo is available in following link
  - https://oss.oetiker.ch/smokeping-demo/?target=rootns

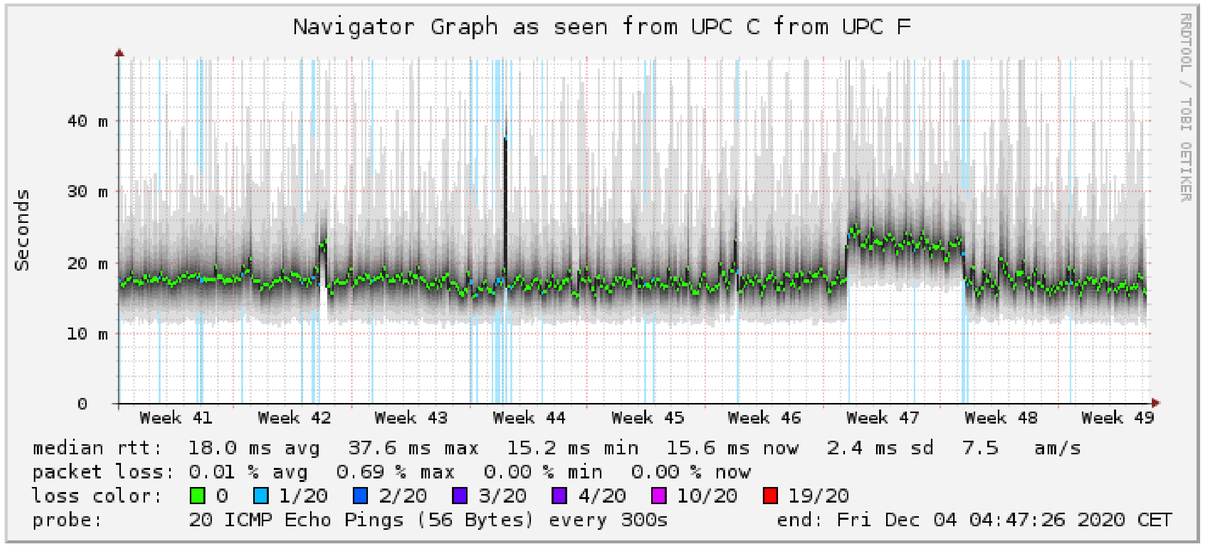






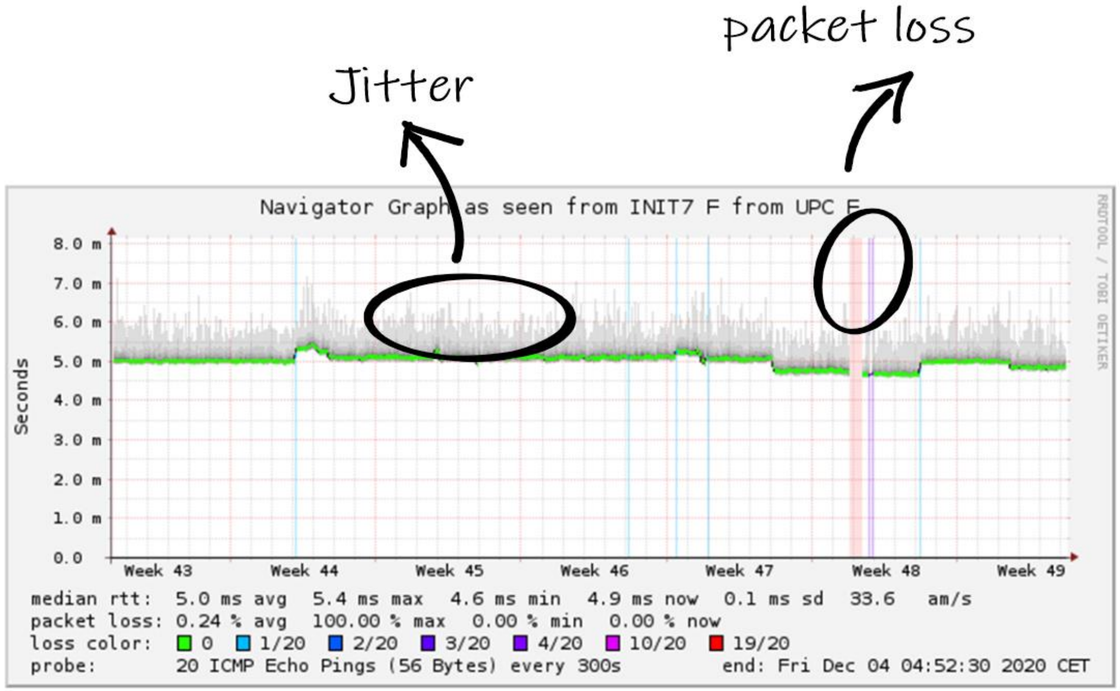
SmokePing - How it works

- Smokeping sends test packets out to the net and measures the amount of time they need to travel from one place to the other and back
- For every round of measurement smokeping sends several packets. It then sorts the different round trip times and selects the median
- The other values are drawn as successively lighter shades of gray in the background (smoke)



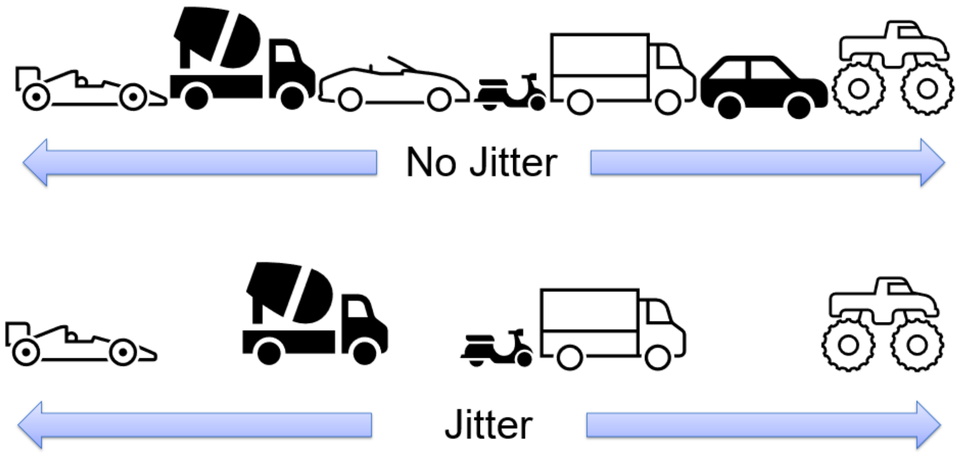
SmokePing - Reading the Graphs

- Heavy fluctuation of the RTT (round trip time) values indicate that the network is overloaded. Also known as jitter
  - This shows on the graph as smoke; the more smoke, the more fluctuation
- Sometimes a test packet is sent out but never returns. This is called packet-loss
  - It can mean that a device in the middle of the link is overloaded or a router configuration somewhere is wrong
- The colour of the median line (horizontal line) changes according to the number of packets lost

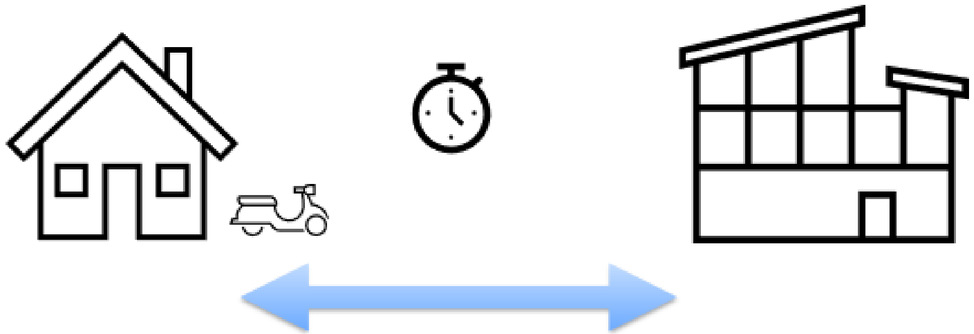


SmokePing - Reading the Graphs

- What is jitter?
  - Irregular time delay in sending of packets.
  - Think of driving a car
  - Depending on the application, jitter could be bad.



- What is latency?
  - Time it takes to get a response or resource
  - How long does it take you to get to work and home again?
  - https://youtu.be/LKmKsXINRAE



SmokePing - Installation

- Install SmokePing in Ubuntu

```
$ sudo apt install smokeping
```

- Docker container
  - https://github.com/dperson/smokeping

SmokePing - Configuration

- Configuration files location
  - `/etc/smokeping/config.d`
  - Configuration files:

```
config.d/
├── Alerts
├── Database
├── General
├── pathnames
├── Presentation
├── Probes
├── Slaves
└── Targets
```

SmokePing - Configuration - General

/etc/smokeping/config.d/General

```
*** General ***

owner    = group10
contact  = group10@apnictraining.net
mailhost = localhost
# NOTE: do not put the Image Cache below cgi-bin
# since all files under cgi-bin will be executed ... this is not good for images.
cgiurl   = http://group10-server.apnictraining.net/smokeping.cgi
# specify this to get syslog logging
syslogfacility = local0
# each probe is now run in its own process
# disable this to revert to the old behaviour
# concurrentprobes = no

@include /etc/smokeping/config.d/pathnames
```

SmokePing - Configuration - Alerts

/etc/smokeping/config.d/Alerts

```
*** Alerts ***
to = noc@apnictraining.net
from = group10@apnictraining.net

+someloss
type = loss
# in percent
pattern = >0%,*12*,>0%,*12*,>0%
comment = loss 3 times  in a row
```

SmokePing - Configuration - Probes

/etc/smokeping/config.d/Probes

```
*** Probes ***

+ FPing
binary = /usr/bin/fping

+ DNS
binary = /usr/bin/dig
lookup = wiki.apnictraining.net
pings = 5
step = 180

+ EchoPingHttp
pings = 5
url = /index.html
```

SmokePing - Configuration - Targets

/etc/smokeping/config.d/Targets

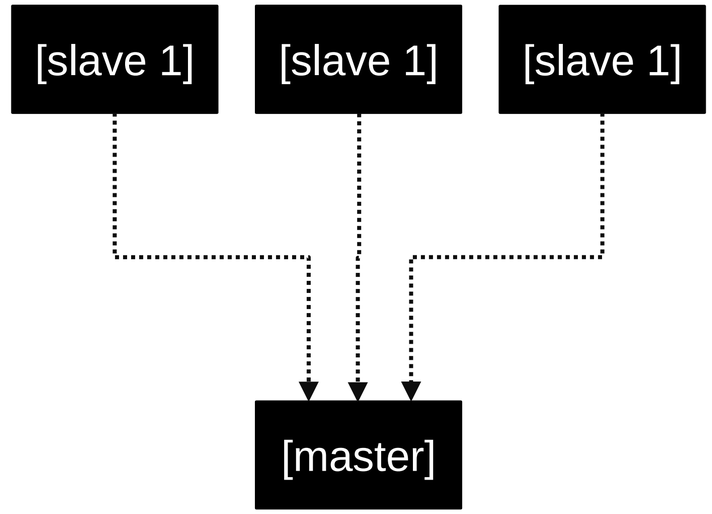
```
*** Targets ***

probe = FPing

menu = Top
title = Network Latency Grapher
remark = Welcome to the SmokePing website of GROUP01. \

+ Local
menu = Local
title = Local Network

++ LocalMachine

menu = Local Machine
title = This host
host = localhost
#alerts = someloss

+ Internet
menu = Internet
title = Internet

++ Google
host = www.google.com
```

SmokePing - Probes

- Different probes are available
  - https://oss.oetiker.ch/smokeping/probe/index.en.html
- Most common probes are
  - FPing
  - DNS
  - EchoPingHttp / EchoPingHttps

SmokePing – Master/Slave

- The Master/Slave concept enables all SmokePing probes to run remotely
- The use case for this is to measure the overall connectivity in a network
- All monitoring data is stored and presented on the Master server, but collected by the slaves
- The slaves will get their configuration information from the master. No need to maintain slaves separately
- The master and the slaves sign their messages by supplying an HMAC-MD5 code or over ssl



Configuration details:
https://oss.oetiker.ch/smokeping/doc/smokeping_master_slave.en.html

SmokePing - Integration

- SmokePing integration with LibreNMS
  - https://docs.librenms.org/Extensions/Smokeping/

SmokePing - References

- SmokePing website:
  - https://oss.oetiker.ch/smokeping/index.en.html
- SmokePing config file:
  - https://oss.oetiker.ch/smokeping/doc/smokeping_config.en.html
- Config examples:
  - https://oss.oetiker.ch/smokeping/doc/smokeping_examples.en.html
- Commandline tool
  - https://oss.oetiker.ch/smokeping/doc/smokeping.en.html
- Demo
  - https://oss.oetiker.ch/smokeping-demo

RRDtool Fundamentals

Tool: Nagios

Nagios

- Nagios is an open source infrastructure monitoring and alerting system
- The free open source part of Nagios, which is known as **Nagios Core**
- There's also a commercial product called **Nagios XI**, which is based on **Nagios Core**

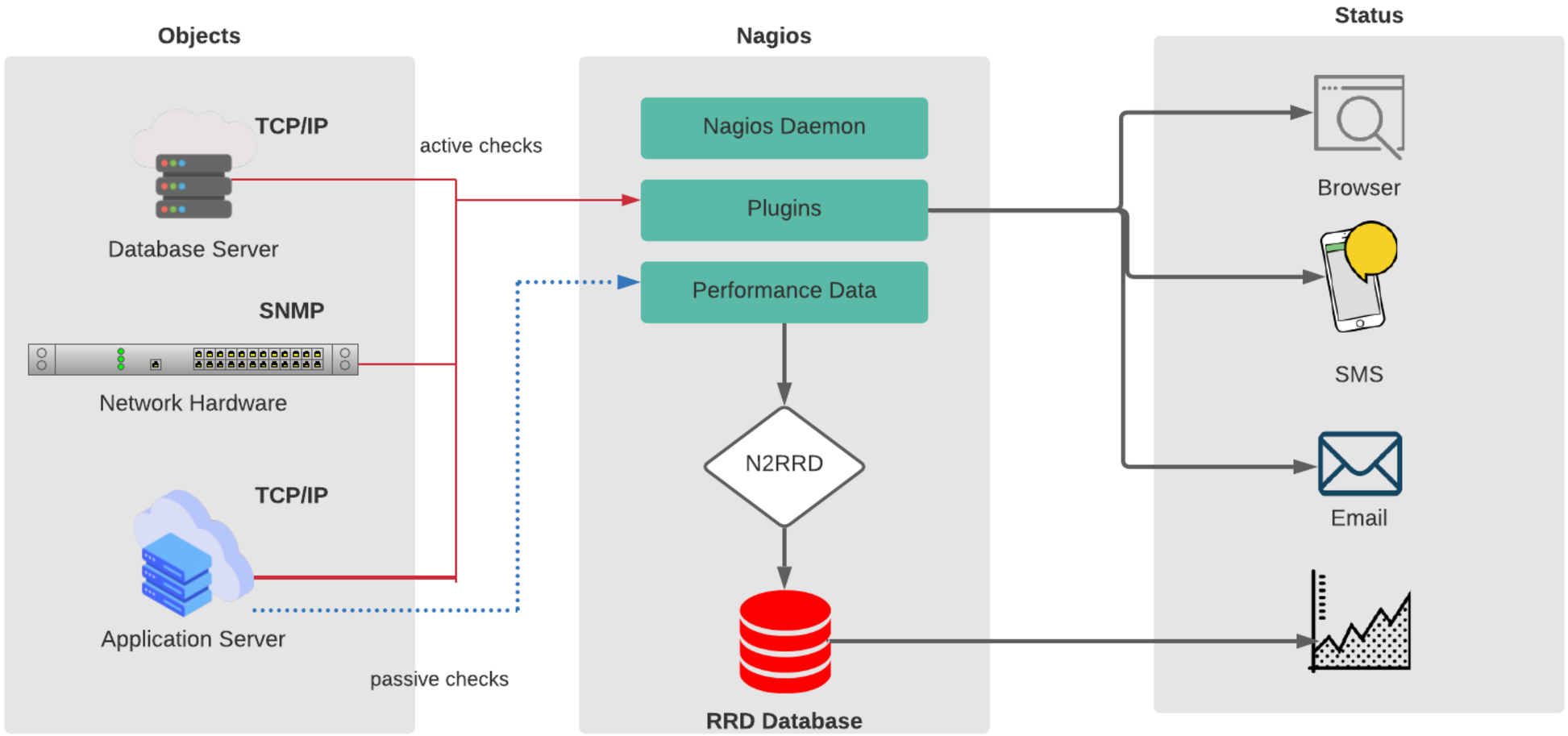
Nagios - Features

- Major features:
  - Monitoring of network services (via SMTP, POP3, HTTP, PING, etc)
  - Monitoring of host resources (processor load, disk usage, etc.)
  - A plugin interface to allow for user-developed service monitoring methods
  - Ability to define network host hierarchy using "parent" hosts, allowing detection of and distinction between hosts that are down and those that are unreachable
  - Notifications when problems occur and get resolved (via email, pager, or user-defined method)
  - Automatic log file rotation/archiving
  - Web interface for viewing current network status, notification and problem history, log file, etc

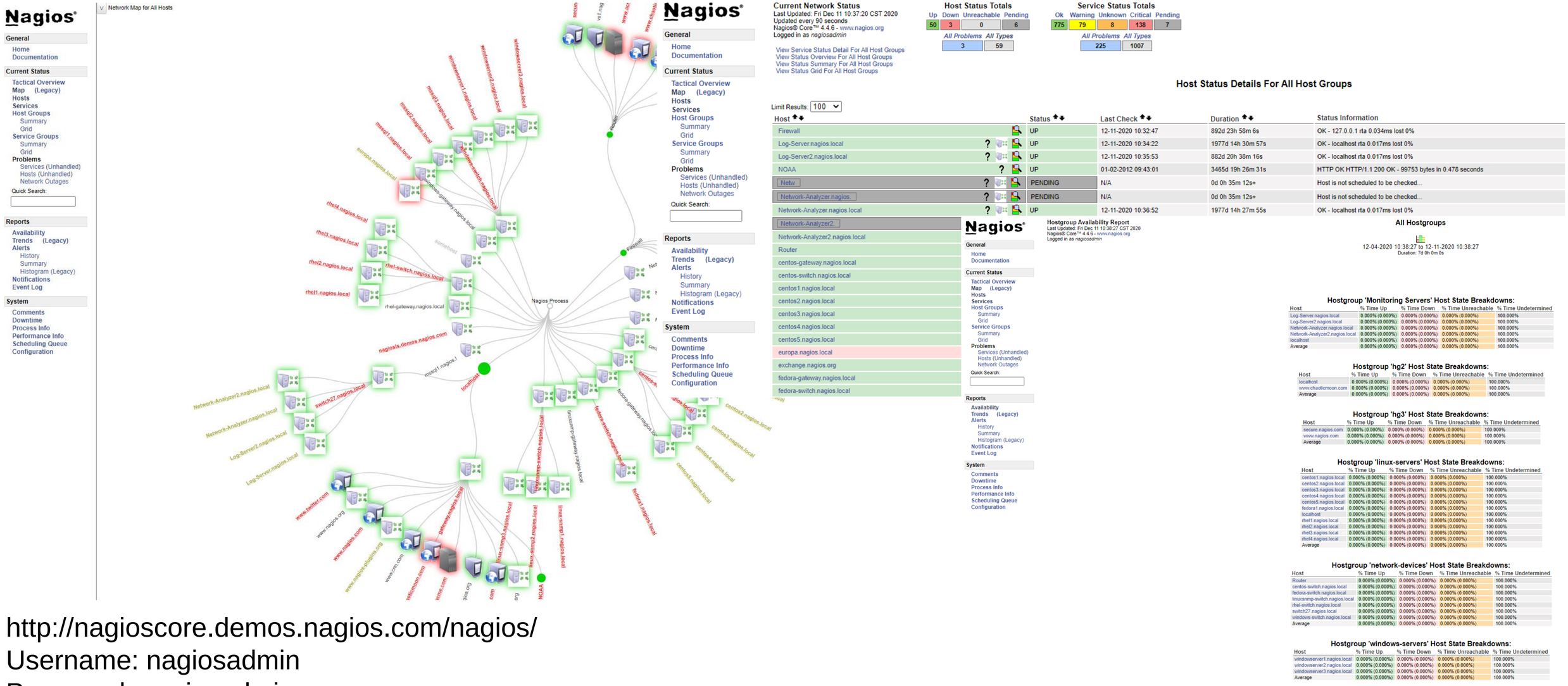
Nagios - Architecture

- Nagios architecture:
  - Nagios has server-agent architecture
  - Nagios server is installed on the host and plugins are installed on the remote hosts/servers which are to be monitored
  - Nagios sends a signal through a process scheduler to run the plugins on the local/remote hosts/servers
  - Plugins collect the data (CPU usage, memory usage etc.) and sends it back to the scheduler
  - Then the process schedules send the notifications to the admin/s and updates Nagios GUI

Nagios - Components



Nagios - Online Demo



http://nagioscore.demos.nagios.com/nagios/
Username: nagiosadmin
Password: nagiosadmin

Nagios - Configuration

- Nagios configuration uses a object oriented representation stored in text files, it does not necessarily require any database
- An object describes a specific unit:
  - Service
  - Host
  - Contact, check command.. with attributes and values
- Main config file
  - **`/usr/local/nagios/etc/nagios.cfg`**

Nagios - Directory Structure

Use	Dir Path
Installation path	`/usr/local/nagios`
Plugins	`/usr/local/nagios/libexec`
Log	`/usr/local/nagios/var/nagios.log`
Run time information	
Config files	`/usr/local/nagios/etc`
Resource files	`/usr/local/nagios/etc/resource.cfg`
WebUI CGI files	`/usr/local/nagios/sbin`

Nagios - Terminology

- Nagios Terminology:
  - Objects
  - Host / Hostgroups
  - Service
  - Commands
  - Contacts

Nagios – Terminology - Objects

- Objects are all elements that are involved in the monitoring and notification logic.
- Type of Nagios objects are
  - Service / Service group
  - Hosts / Host group
  - Contacts / Contact group
  - Commands
  - Timeperiod
  - Notification escalations
  - Notificatin dependencies
- Objects can be defined in .cfg files or in directories @ /usr/local/nagios/etc/objects/

Nagios – Terminology - Host

- A host is anything with an IP, URL, or FQDN
  - such as a physical server, a VM host or guest, a router, or a webpage
- Hostgroups enable us to logically group sets of hosts for display, configuration and reporting purpose
- A service have to be linked to an host
- Only UP and DOWN states

Nagios – Terminology - Host

host.cfg

```
define host {
                use                             generic-host
                host_name                       canireachthe.net
                alias                           canireachthe.net
                address                         canireachthe.net
                max_check_attempts              5
                check_period                    24x7
                notification_interval           30
                notification_period             24x7
}
```

Nagios – Terminology - Service

- A granular metric being monitored on a host
  - such as CPU and memory usage, drivespace, interface bandwidth, or the status of a system service or process.
- Servicegroups enable you to logically group sets of services for display and reporting purposes

Nagios – Terminology - Service

service.cfg

```
define service {
    use                 generic-service
    host_name           canireachthe.net
    service_description HTTP
    check_command       check_http
}
```

Nagios – Terminology - Commands

- To execute check plugins or to send user notifications
  - Links Nagios attributes found in definitions (ex: host name) to plugin parameters
  - Already provided for most common plugins (see objects/commands.cfg file)
  - Basic wrapping to send emails (objects/commands.cfg file)

Nagios – Terminology - Commands

objects/commands.cfg

```
##############################################################################
#
# SAMPLE NOTIFICATION COMMANDS
#
# These are some example notification commands.  They may or may not work on
# your system without modification.  As an example, some systems will require
# you to use "/usr/bin/mailx" instead of "/usr/bin/mail" in the commands below.
#
##############################################################################

define command {

    command_name    notify-host-by-email
    command_line    /usr/bin/printf "%b" "***** Nagios *****\n\nNotification Type:
$NOTIFICATIONTYPE$\nHost: $HOSTNAME$\
nState: $HOSTSTATE$\nAddress: $HOSTADDRESS$\nInfo: $HOSTOUTPUT$\n\nDate/Time:
$LONGDATETIME$\n" | /usr/sbin/sendmail -s
"** $NOTIFICATIONTYPE$ Host Alert: $HOSTNAME$ is $HOSTSTATE$ **" $CONTACTEMAIL$
}
```

Nagios – Terminology - Contacts

- To specify who to notify, how to notify and when to notify
- A notification is an email or text that is sent when state changes are detected on monitored objects

Nagios – Terminology - Contacts

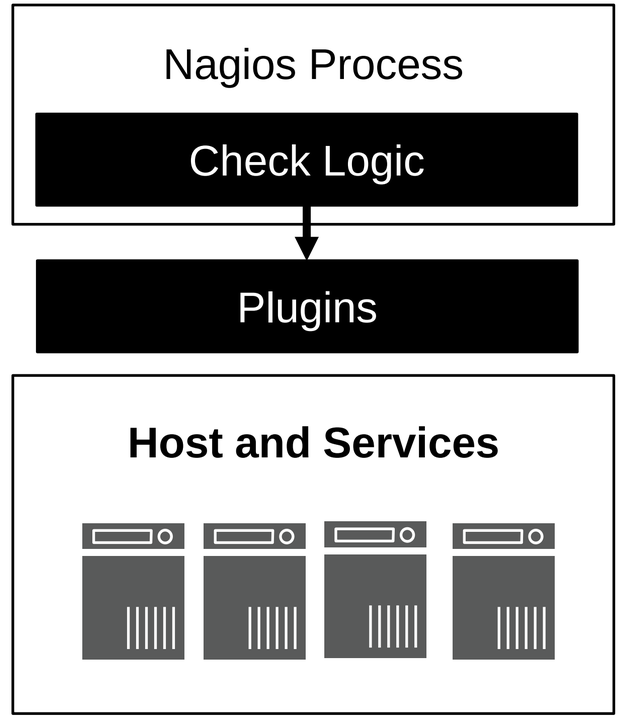
objects/contacts.cfg

```
##############################################################################
#
# CONTACTS
#
##############################################################################

# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the
# 'generic-contact' template which is defined elsewhere.

define contact {

    contact_name            nagiosadmin
    use                     generic-contact
    alias                   NOC ADMIN
    email                   noc@example.com
}
```

Nagios - How Checks Work

- Nagios execute plugins on a regular scheduled basis to check the operational state of the host or service
- Response Codes:

Plugin Return Code	Service State	Host State
0	OK	UP
1	WARNING	UP or DOWN/UNRECHABLE
2	CRITICAL	DOWN/UNRECHABLE
3	UNKNOWN	DOWN/UNRECHABLE



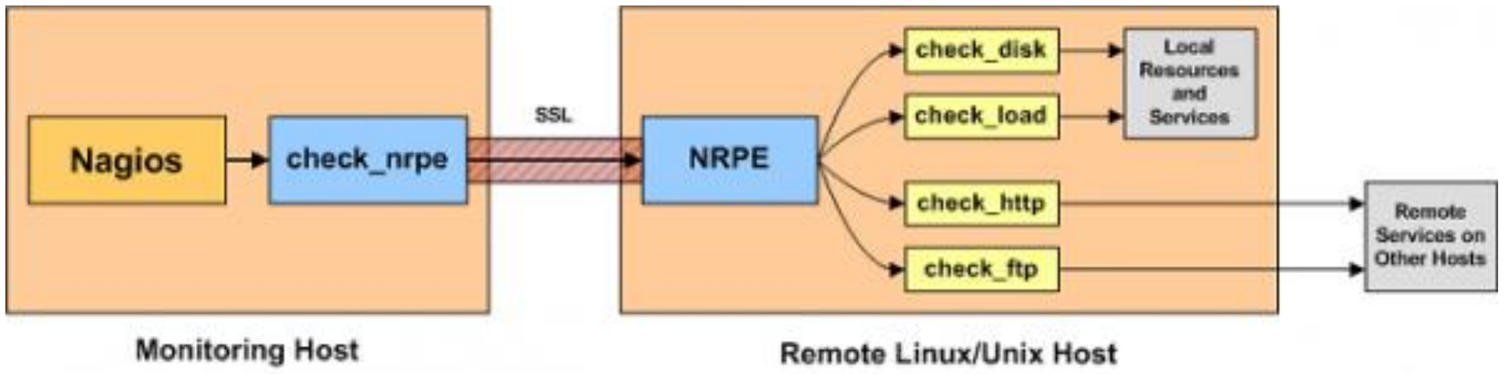
- Store the status to determine if they are working properly or not
- Two states:
  - Soft state -> When a host or service is down for a very short duration of time and its status is not known or different from previous one
  - Hard state -> When max_check_attempts is executed and status of the host or service is still not OK

Nagios - Plugins

- Plugins helps to monitor databases, operating systems, applications, network equipment, protocols with Nagios
- Types of Nagios Plugins:
  - **Official Nagios Plugins** : There are 50 official Nagios Plugins. Official Nagios plugins are developed and maintained by the official Nagios Plugins Team
  - **Community Plugins** : There are over 3,000 third party Nagios plugins that have been developed by hundreds of Nagios community members
  - **Custom Plugins** : You can also write your own Custom Plugins. There are certain guidelines that must be followed to write Custom Plugins
- Community Plugins:
  - https://exchange.nagios.org/

Nagios - Nagios Remote Plugin Executor

- NRPE (Nagios Remote Plugin Executor) allows us to remotely execute Nagios plugins on other Linux/Unix machines
- This allows us to monitor remote machine metrics (disk usage, CPU load, etc.)
- NRPE can also communicate with some of the Windows agent addons, check metrics on remote Windows machines as well!



NRPE: shorturl.at/bLUV8

Nagios - Summary

- Open source monitoring solution
- Based on simple concepts: checks, states, notifications
- Easily extensible and integrable
- No discovery mechanism
- Experience it during lab exercises!
  - Monitoring hosts and their services
  - Developing and testing our own plugin
  - Experimenting state transitions and notifications

Nagios - References

- Nagios website:
  - https://www.nagios.org/
- Nagios Tutorial:
  - https://www.tutorialspoint.com/nagios/index.htm
- Demo
  - http://nagioscore.demos.nagios.com/nagios/
- Nagios Certification
  - https://www.nagios.com/services/certification/

RRDtool Fundamentals

Module 2: LAB

Labs

- Please follow the lab modules for
  - Lab 1: SmokePing
  - Lab 2: Nagios

Thank You!