An Introduction to Nagios

Ronald Starink (Nikhef)

EGEE08, Istanbul 24-Sep-2008

www.eu-egee.org

e-infrastructure

Outline

- **What is Nagios?**
- **Getting Started**
  - Installation & Configuration
  - Object Definitions
- **More Advanced Setups**
  - Remote Checks
  - Organizing Object Definition
- **Summary**

What is Nagios?

- **“Nagios® is an Open Source host, service and network monitoring program”**

  www.nagios.org

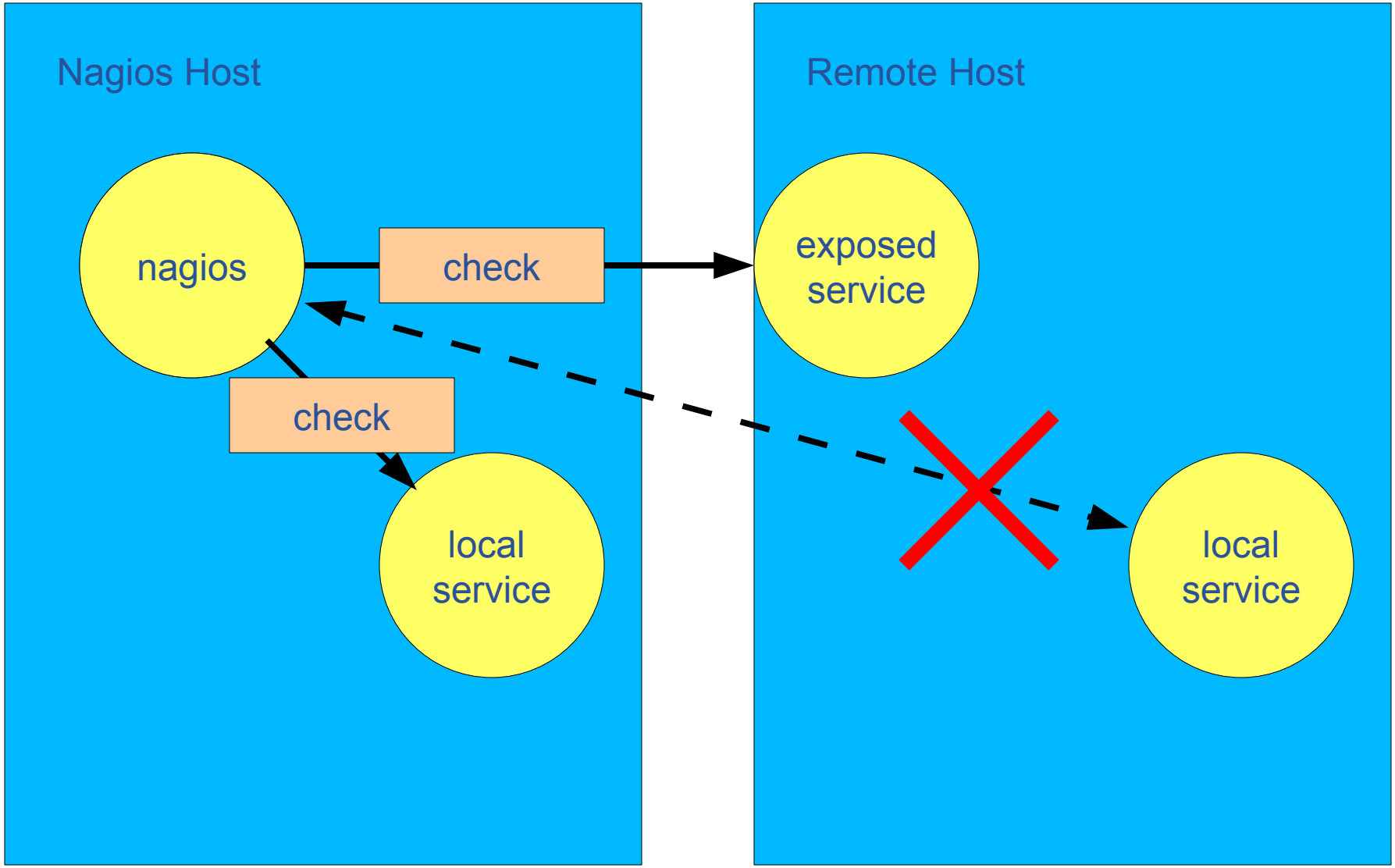
- **“Nagios is a popular open source computer system and network monitoring application software. It watches hosts and services, alerting users when things go wrong and again when they get better.”**

  www.wikipedia.org

Architecture

- **Simplest setup:**
  - A central server running Nagios daemon
  - That runs local check scripts
  - To determine the status of services on that host and remote hosts
- **A host is a computer running on the network which runs one or more services to be checked**
- **A service is anything on the host that you want checked. Its state can be one of: OK, Warning, Critical or Unknown**
- **A check is a script run on the server whose exit status determines the state of the service: 0, 1, 2 or 3**

Simplest Nagios Setup



Installation

- **Hardware: standard server**
  - E.g. Pentium-4 3 GHz, 2 GB RAM, 60 GB disk
- **Nagios RPMs for RHEL available from the DAG repository**
  - nagios – the main server software and web scripts
  - nagios-plugins – the common set of check scripts used to query services
  - nagios-nrpe – Nagios Remote Plugin Executor
  - nagios-nsca – Nagios Service Check Acceptor
- **Setup:**
  - Install the RPMs
  - Configure the web server
  - Edit the config files to suit your setup

Nagios Configuration

- **Main configuration file**
  - Functionality / options
  - Location of directories or files
- **CGI configuration**
- **Macro files**
  - User variables
    - `$USER1$=/usr/lib/nagios/plugins`
- **Object definition files**
  - What is monitored (hosts, services)?
  - How is it monitored (invocation script)?
  - Who should be notified in case of problems, when, how?
  - Use groups and templates to organize setup
    - When your setup grows
    - Things can get complex!

Main Configuration

- **Customization of default file**
  - `/etc/nagios/nagios.cfg`
- **File contains comments describing options**
  - But read the manual!
- **ALWAYS test configuration before (re)starting**
  - `nagios -v /etc/nagios/nagios.cfg`
  - Or you're in for surprises

Object Definition - Hosts

```
define host{
        host_name                       my-host
        alias                           my-host.domain.nl
        address                         192.168.0.1
        check_command                   check-host-alive
        max_check_attempts              10
        check_period                    24x7
        notification_interval           120
        notification_period             24x7
        notification_options            d,r
        contact_groups                  unix-admins
        register                        1
}
```

Object Definition - Services

```
define service{
        name                            ping-service
        service_description             PING
        is_volatile                     0
        check_period                    24x7
        max_check_attempts              4
        normal_check_interval           5
        retry_check_interval            1
        contact_groups                  unix-admins
        notification_options            w,u,c,r
        notification_interval           960
        notification_period             24x7
        check_command    check_ping!100.0,20%!500.0,60%
        hosts                           my-host
        register                        1
}
```

Object Definition - Command

- **Commands wrap the check scripts**

```
define command{
        command_name    check-host-alive
        command_line    $USER1$/check_ping -H     $HOSTADDRESS$ -w
  99,99% -c 100,100% -p 1
        }
```

- **and the alerts**

```
define command{
        command_name    notify-by-email
        command_line    /usr/bin/printf "%b" "***** Nagios  *****\
  n\nNotification Type: $NOTIFICATIONTYPE$\n\nService:
  $SERVICEDESC$\nHost: $HOSTALIAS$\nAddress: $HOSTADDRESS$\nState:
  $SERVICESTATE$\n\nDate/Time: $LONGDATETIME$\n\nAdditional Info:\
  n\n$SERVICEOUTPUT$" | /bin/mail -s "** $NOTIFICATIONTYPE$ alert
  - $HOSTALIAS$/$SERVICEDESC$ is $SERVICESTATE$ **" $CONTACTEMAIL$
        }
```

Check Scripts

- **Standard nagios-plugins rpm:**
  - check_load, check_disk
  - check_ldap, check_mysql
  - And many others!
- **Writing your own check scripts is easy, can be in any language.**
  - Active scripts just need to set the exit status and output a single line of text
  - Passive checks just write a single line to the server's command file

Object Definition – Contacts

- **Contacts are the people who receive the alerts:**

```
define contact{
        contact_name                    ronald
        alias                           Ronald Starink
        service_notification_period     24x7
        host_notification_period        24x7
        service_notification_options    w,u,c,r
        host_notification_options       d,r
        service_notification_commands   notify-by-email
        host_notification_commands      host-notify-by-email
        email                           someone@somewhere
}
```

- **Contactgroups group contacts:**

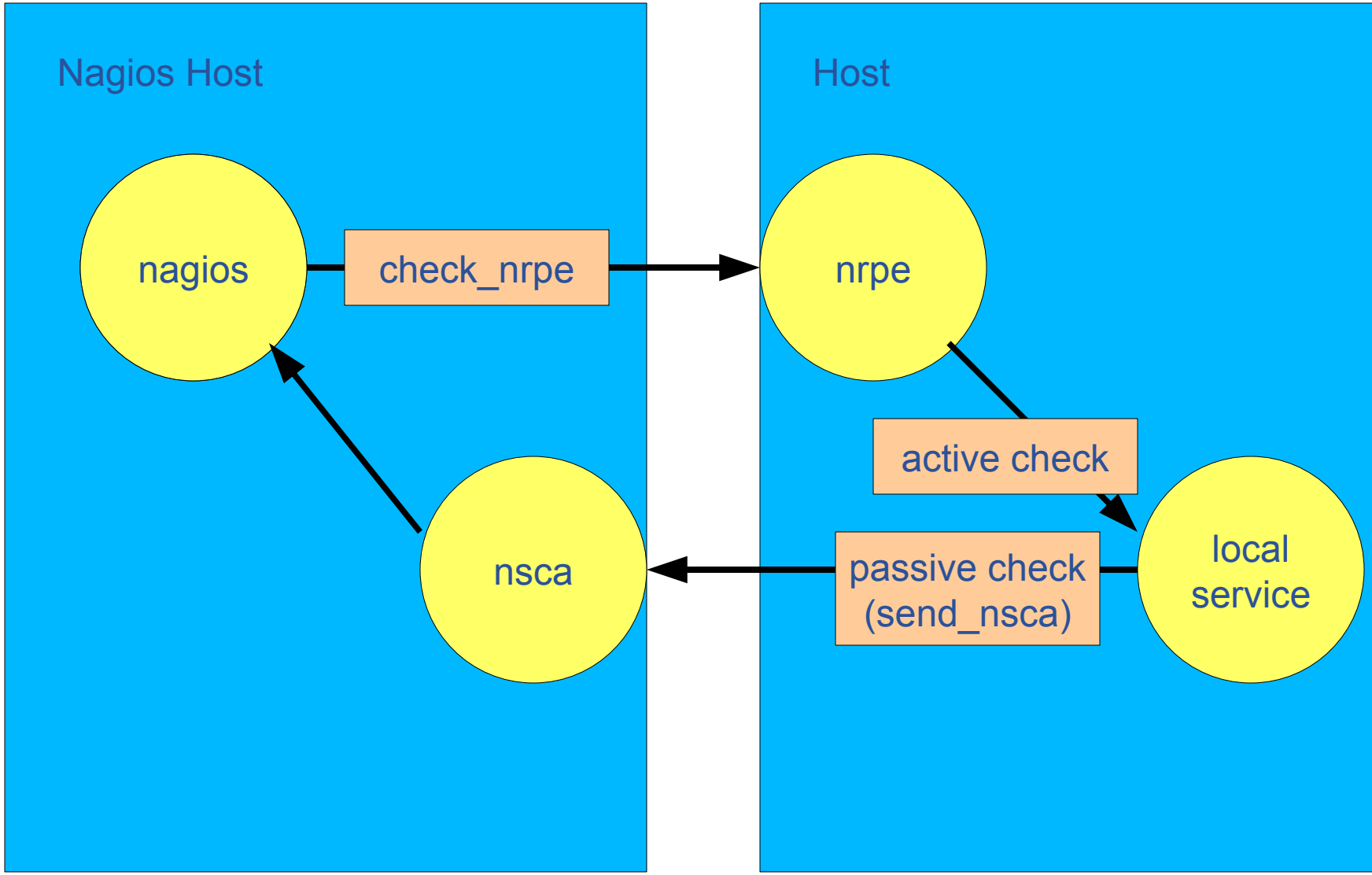
```
define contactgroup{
        contactgroup_name       unix-admins
        alias                   Unix Administrators
        members                 ronald
}
```

Object Definition – Time Period

- **Time periods define when things, checks or alerts, happen:**

```
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
}
```

Remote Checks



Remote Checks - NRPE

- **NRPE:**
  - a daemon that runs on a remote host to be checked
  - a corresponding check script on the Nagios server
- **Nagios daemon runs the check_nrpe script, which contacts the NRPE daemon, which runs the check script locally and returns the output:**

Nrpe.cfg (on remote host):

```
command[check_load]=/usr/lib/nagios/plugins/check_load -w 15,10,5 -c
   30,25,20
```

Nagios.cfg (on Master server):

```
define command{
        command_name    check_nrpe_load
        command_line    $USER1$/check_nrpe -H $HOSTADDRESS$ -c
   check_load
}
```

- **Check script runs under non-root account**
  - May have to use sudo to perform certain checks

NRPE configuration

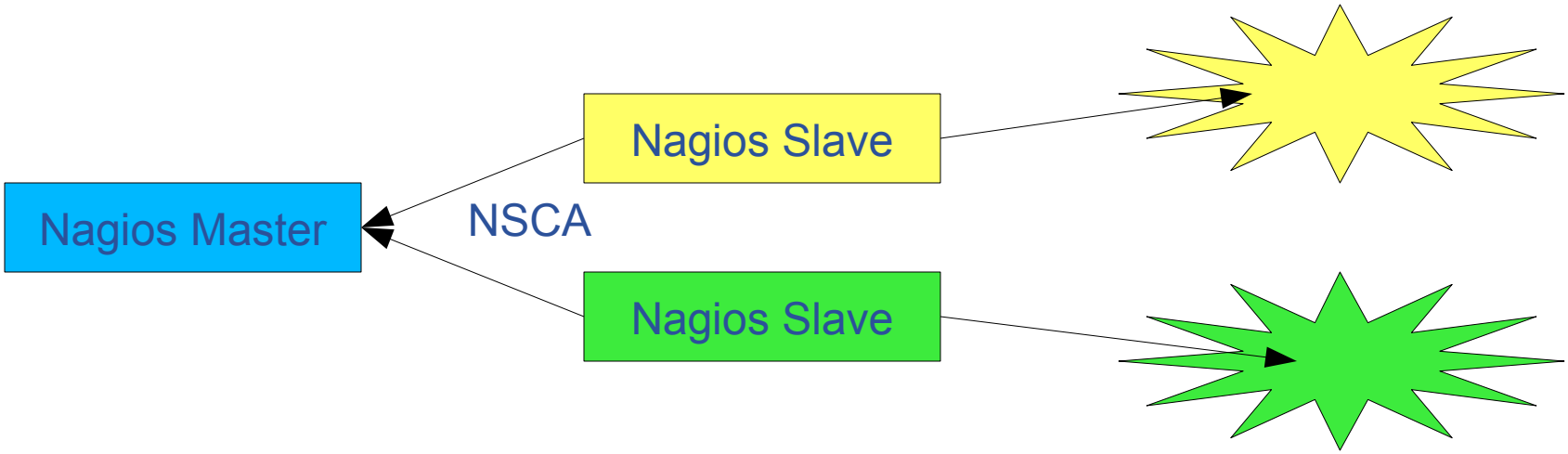
```
pid_file=/var/run/nrpe.pid
server_port=5666
nrpe_user=nrpe
nrpe_group=nrpe
dont_blame_nrpe=0
debug=0
command_timeout=60
command[check_users]=/usr/lib64/nagios/plugins/check_users -w
   5  -c 10
command[check_zombie_procs]=/usr/.../check_procs -w 5 -c 10 -s
   Z
command[check_load]=/usr/.../check_load -w 30,20,10 -c
   45,30,20
command[check_total_procs]=/usr/.../check_procs -w 300 -c 400
command[check_system_disk]=/usr/.../check_disk -w 20% -c 10%
    -p /dev/sda2
command[check_scratch_disk]=/usr/.../check_disk -w 20% -c 10%
    -p /dev/sda5
```

Active vs Passive Checks

- **For some services running a script to check their state every few minutes (so called active checking) is not the best way.**
  - Service has its own internal monitoring
  - One script can efficiently check the status of multiple related services
- **The nagios service can be set to read “commands” from a named pipe**
  - Any process can then write in a line updating the status of a service (passive check)
  - Web frontend’s cgi script can also write commands to the file to disable checks or notifications for a host or service for example.

Remote Checks - NSCA

- **NSCA, is a script/daemon pair that allow remote hosts to run passive checks and write the results into that nagios server's command file.**
  - Check script on remote host uses script send_nsca to forward the result to the nsca daemon on the server, which writes the result into the command file
  - Check may be initiated by cron
    - Use random delay not to overflow NSCA!
  - Can be used with event handlers to produce a hierarchy of Nagios servers



Organizing – Host/Service Groups

- **Host and service groups let you group together similar hosts and services:**

```
define hostgroup{
        hostgroup_name  WN
        alias           Worker Nodes
}
define servicegroup{
        servicegroup_name       topgrid
        alias                   Top Grid Services
}
```

- **Plus a hostgroups or a servicegroups line in the host or service definition**

Organizing - Templates

- **You can define templates to make specifying hosts and services easier:**

```
define host{
        name                    generic-unix-host
        use                     generic-host
        check_command           check-host-alive
        max_check_attempts      10
        check_period            24x7
        notification_interval   120
        notification_period     24x7
        notification_options    d,r
        contact_groups          unix-admins
        register                0
}
```

- **Reduces a host definition to:**

```
define host{
        use                     generic-grid-frontend-host
        host_name               my-host
        alias                   my-host.domain.nl
        address                 192.168.0.1
}
```

Going Further - 1

- **Service Hierarchies, services and hosts can depend on other services or hosts so for instance:**
  - If the web server is down don't tell me the web is unreachable
  - If the switch is down don't send alerts for the hosts behind it
- **Event Handlers:**
  - Nagios can attempt to rectify the fault by running an eventhandler
- **Hierarchy of Nagios servers**
  - 1 Master
    - Runs the web frontend
    - Receives test results from slave servers via NSCA
  - *N* Slaves
    - Schedule checks on a specific cluster

Going Further - 2

- The cgi scripts, templates and style sheets that build the web pages can be edited to add extra information
- Nagios has a myriad of other features not touched upon here from state stalking to flap detection, notification escalations to scheduling network, host or service downtimes

Summary

- **Nagios is a very useful and flexible tool for sysadmins to know the state of their systems**
- **It may seem intimidating when you first look at it**
- **Advice:**
  - Install it on your test node (though this may well end up as your master server)
  - Run a few check scripts by hand to get the feel for them
  - Set up a simple config file that runs a few checks on the local host
  - Install nrpe on the host and nrpe and nagios-plugins on a remote host
  - Run check nrpe by hand to get it working, then add a couple of simple checks on the remote host
  - NOW THINK ABOUT HOW YOU WANT TO ORGANIZE YOUR CONFIG FILES
  - Now add hosts and service until you run out, then write some more
  - Make backups of your configurations, store them under revision control, use Quattor ;-)