Monitoring with Nagios and graphing with PerfParse

Garrett Honeycutt
2005-10-05
What is Nagios?

- Open Source (GPL) monitoring
- Runs on *nix
- Monitors hosts, services, and anything else
- Provides a status overview (dashboard)
- Sends notifications
- Triggers events
- Over 150,000 downloads of the stable 1.2 branch
Why monitor?

- Know about an issue before it causes problems
- Know before someone complains
- Be proactive about problems (then automate)
- Performance metrics
- More time for projects instead of putting out fires
- Prove uptime (SLA's, raises, etc)
Architecture

- Modular design
- Nagios daemon contains monitoring logic and task coordination
- CGI's allow users to view status and submit commands
- External Plugins handle the actual monitoring
- External commands can be triggered to manage alerts or take action
- Easy to integrate with 3rd party apps
Nagios Overview

• Hosts
  - Usually a physical object (server, switch, routers, printers, etc.)
  - Can have parent/child relationships with other hosts
  - Provide one or more services

• Services
  - Things associated with or provided by a host
  - Tangible services (disk usage, printer toner supply)
  - Intangible services (HTTP, SMTP, IMAP, DNS)
Nagios Overview

• Example: Small Network Layout
Nagios Overview

- Hosts as viewed by Nagios
Nagios Overview

- Services as viewed by Nagios
Host Checks

- Hosts
  - “Containers” of services
  - Three states: UP, DOWN, UNREACHABLE

- Host Checks
  - Checked with plugins like services (usually icmp)
  - Checks are performed on-demand after service state changes
    - Critical decision point: is the host or the service the real problem?
  - Can trigger a route verification
Host Route Verification

- Why might a host not be UP?
  - The host is DOWN
  - The route to the host is blocked by one or more other hosts (UNREACHABLE)

- Route verification
  - Determines whether hosts are DOWN or UNREACHABLE
  - Can be very time-intensive if network problems are widespread

- Why is it useful?
  - Helpful in determining the “real” cause of widespread problems
  - Suppressing a flood of notifications
Host Route Verification

- Logical host relationships - The world according to Nagios
Host Route Verification

- A problem is detected with 'Port Status' service on Switch2...
Host Route Verification

- Switch2 is checked for problems and found to NOT be UP...
Host Route Verification

- Check is propagated upstream to Firewall1, which is also NOT UP...
Host Route Verification

- Check is propagated upstream to Switch1, which IS found to be UP
Host Route Verification

- Reachability of Switch2 can now be determined - Firewall1 is DOWN and Switch2 is UNREACHABLE.
Host Route Verification

- Checks are propagated to children of SWITCH2, which are found to be UNREACHABLE
Host Route Verification

- Other children of Firewall1 are checked later, found to be UNREACHABLE
Service Checks

• Services
  – Services are why we monitor

• Service checks
  – Checks are performed at regularly scheduled intervals using plugins
  – Multiple checks may be run in parallel
  – Four possible states: OK, WARNING, CRITICAL, and UNKNOWN
  – You can monitor anything you can write a plugin for
Plugins

- How are hosts and services monitored?
  - Nagios doesn't understand network addresses, protocols, or services
  - Nagios passes information about what needs to be checked to external commands (plugins)
  - Plugins perform the actual checks of hosts and services and return information back to Nagios
Plugins

- Coded in C, PHP, Perl, Python, shell scripts, etc
- Return error codes and hopefully performance data
- Official plugins available on nagios.org
- NagiosExchange is the place to share plugins
- ANYTHING that can output text can be fed into nagios for alerts and graphed.
Plugins

- Uses exit codes to show state status
  - 0 = OK
  - 1 = Warning
  - 2 = Critical
  - 3 = Unknown

[ root @ nagi os ] # check_mem -w 80 -c 95
OK:  19% Used Memory | MemUsed=19%; 80; 95

[ root @ nagi os ] # echo $? 
0
Service Checks

- Active
  - synchronous
  - scheduled and initiated by nagios

- Passive
  - asynchronous
  - SNMP traps
  - security events
  - Distributed monitoring
Plugins

- Local plugins
  - ran from host being checked
    - check_mem
    - check_disk
    - check_procs

- Remote plugins
  - ran from nagios server
    - check_icmp
    - check_http
    - check_tcp
Local checks on Remote Hosts

- NRPE
  - runs as a daemon on the nagios server
  - client ran via xinetd/inetd
  - encrypted
  - runs local checks

- NSClient / check_nt
  - NSClient runs as service on windows host
  - check_nt performs checks from nagios server
  - encrypted (needs mcrypt libs)
NSCA

- sends passive checks
- nsca daemon runs on nagios server
- send_nsca sends encrypted data to nsca on nagios server
- Used with asynchronous (or non nagios scheduled) events
  - SNMP Traps
  - Security events
  - Distributed Monitoring
Something is on fire, what now?

- Notifications
  - who to notify?

- Scheduled Downtime
  - are we in it? has it started?

- Event Handlers
Notifications

- When do they occur?
  - At first occurrence of a problem
  - Each \( x \) minutes during continued problem (can be annoying)
  - During changes between problem states
  - At time of recovery
  - At start and stop of flapping

- Who do they get sent to?
  - By default, to primary contacts for the host or service
  - Can be escalated to different contacts, depending on:
    - Current state of host or service
    - Current notification number
    - Timeperiod (day of week, time of day)
Notifications

- How are notifications sent?
  - Nagios executes a user-defined command
  - Any shell script or executable can be used
  - Alert information provided via macros and environment vars
- Some potential notification methods:
  - Email
  - Pager
  - SMS
  - Instant Messages (IM, Jabber, ICQ)
  - Audible
  - Visual alerts (billboards, indicator lights)
Event Handlers

- If you have a problem, let nagios solve it
- Simply add the event handler to services.cfg
- You can pass it environment variables such as $SERVICESTATE$ $SERVICEATTEMPT$ which can be used by your script in a case statement.
- Can be run on the remote machine via nrpe or have the nagios server ssh out with predefined keys
Configuration Files

- **Runtime Configuration**
  - cgi config (cgi.cfg)
  - daemon config (nagios.cfg)
- **Basic Objects**
  - check commands / plugin syntax (checkcommands.cfg)
  - contacts (contacts.cfg)
  - hosts (hosts.cfg)
  - services (services.cfg)
- **Advanced Objects**
  - host groups (hostgroups.cfg)
  - contact groups (contactgroups.cfg)
  - escalations (escalations.cfg)
  - time periods (timeperiods.cfg)
  - PerfParse (perfparses.cfg)
nagios.cfg

cfg_file = $NAGPATH$/etc/perfparse.cgi

process_performance_data = 1

xpdf_file_service_perfdata_file = $NAGPATH$/var/rw/perfdata-service.log

• this is a named pipe owned by nagios:nagioscmd

host_perfdata_command = process-host-perfdata

service_perfdata_command = process-service-perfdata
contacts.cfg

- Whom, When, Why, and How to alert people

```bash
define contact {
    contact_name                garrett
    alias                       Garrett Honeycutt
    service_notification_period 24x7
    host_notification_period    24x7
    service_notification_options w, u, c, r
    host_notification_options   d, u, r
    service_notification_commands email, pager
    host_notification_commands  host-email, host-pager
    email                       fire@garretthoneycutt.com
    pager                       317-xxx-xxxx@pager.com
}
```
hosts.cfg

- entities out there that provide the services we care about

define host {
    host_name web1.foo.bar
    alias web1
    parents switch1
    address 10.10.10.2
    notification_enabled 1
    check_command check-host-alive
    max_check_attempts 10
    notification_interval 60
    notification_period 24x7
    notification_options d,u,r
}
checkcommands.cfg

- Links the plugins to usable config objects
- uses environment variables

```bash
# check_sslcert
define command{
    command_name check_sslcert
    command_line $USER1$/check_http -H $HOSTADDRESS$ -S -C $ARG1$
}
```
services.cfg

- Resources provided by our hosts

```c
define service {
    name SSL_Cert
    notification_options w, u, c, r
    notification_interval 60
    check_period 24x7
    max_check_attempts 3
    normal_check_interval 3
    retry_check_interval 1
    host_name web1
    contacts garrett
    check_command check_sslcert!28
}
```
PerfParse

- Written in C
- Stores info from named pipe into MySQL
- Great for trending and making predictions
- Only one graph at a time
- Only one metric per graph
- Extremely easy to compile and setup
Performance Data

- Plugins should have it. If not it is easy to add.
- One line of text output, can return multiple metrics
- Use the same return codes
- 'label'=value[UOM];[warn];[crit];[min];[max]

```
[root@nagios]# check_mem -w 80 -c 95
OK: 33 % Used Memory | MemUsed = 33 %; 80; 95
```
PerfParse Sample

Graph showing resource utilization over time for a host named 'dib.3gupload.com', service 'HTTPD processes', and metric 'Processes'. The graph indicates fluctuations in resource usage with peaks and troughs, marked by different color bands for value, warning, and critical thresholds.
Other Resources

- http://nagios.org
- http://nagiosexchange.org
- Nagiosplug-help mailing list
- gh@garretthoneycutt.com

Questions?
Nagios

CINLUG
Central Indiana Linux Users Group
www.cinlug.org