Best Practices in Monitoring

Lars Vogdt
Team Lead SUSE DevOPS
<Lars.Vogdt@suse.com>

About Lars Vogdt

- Co-developer of the SUSE School Server (2003)
- Team lead openSUSE Education since 2006
- Team lead internal IT Services Team 2009 – 2016
- Team lead DevOPS Team since Sep. 2016 (Main Target: Build Service)
  - Responsible for Product Generation, Build Service and Package Hub inside and outside SUSE
- Responsible for “monitoring packages” at SUSE

Control your infrastructure

Optimize your IT resources

?

How can you do that without knowing your requirements and your current resources

?

Conclusion:

Monitoring is a basic requirement before thinking about anything else...

Agenda

SUSE monitoring packages

Tips and Tricks

- Generic Tips
- Examples

High available and/or load balanced monitoring: one possible way to go

Demos:

- Icinga, PNP4Nagios, NagVis
- automatic inventory via check_mk
- Pacemaker / Corosync (SUSE Linux Enterprise High Availability)
- (mod_)Gearman
- Salt
- …

The future of monitoring @SUSE

SUSE monitoring packages

SUSE monitoring packages

Official vs. unsupported

Official supported
SUSE official repos
Nagios for <= SLES 11
nagios-plugins <= SLES 11
Icinga 1 for >= SLES 12 via SUSE Manager
monitoring-plugins for >= SLES 12

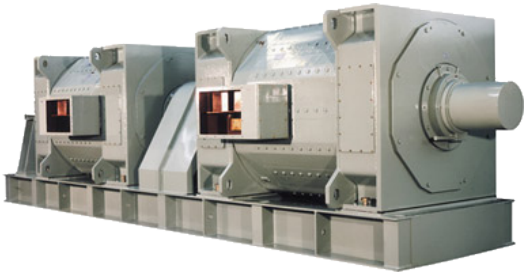
Tips and Tricks

Monitoring?

1. Monitoring starts before a machine/service goes into production
2. Monitoring without history will not help to think about the future
3. Monitoring without graphs and trends is hard to understand
4. Monitoring should be easy

Monitoring starts: early

What can be monitored



- **SPS monitoring** (see **http://snap7.sourceforge.net/**) **?**
- **check weight and temperature of your bees ?**

- **check your coffee mug ?**
- **check for housebreakers ?**

- **monitor what should be there or what is there ?**
- check, if a host does what is configured in CMDB ?

→ Use monitoring to ensure that services and states match your desired model

What can be checked?

Nearly everything is possible!

Minimal requirements listed below:

Your script returns one of the following Exit-Codes:

3 : Unknown – something outside the normal control range (of your script?) happened

2 : Something critical happend! Help needed!

1 : well, it works currently – but be warned

0 : everything ok

Some (human readable) output on STDOUT would be nice, but is not necessary for Nagios or Icinga itself.

Print performance data on STDOUT, separated from normal output via '|'
https://nagios-plugins.org/doc/guidelines.html.

The above is true not only for Nagios or Icinga, other monitoring systems like Zabbix, Centreon, op5, server density and a lot more are at least compatible.

Example check: check_file_exists

```
#!/bin/bash
# Check if a local file exist
while getopts F: VAR; do
    case "$VAR" in
        F ) LOCAL_FILE="$OPTARG" ;;
        * ) echo "wrong syntax: use $o -F <file to check>"
            exit 3 ;;
        esac
done
if test -e "$LOCAL_FILE"; then
    if test -x "$LOCAL_FILE"; then
        echo "Critical: $LOCAL_FILE exists and is executable"
        # Nagios exit code 2 = status CRITICAL = red
        exit 2
    else
        echo "Warning: $LOCAL_FILE exists"
        # Nagios exit code 1 = status WARNING = yellow
        exit 1
    fi
else
    echo "OK: $LOCAL_FILE does not exist"
    # Nagios exit code 0 = status OK = green
    exit 0
fi
```

Eventhandlers

If a service or host is in a defined, unwanted state, trigger external scripts to “solve” the problem automatically.

(Restart apache if it crashes, send SMS if nobody acknowledges a problem, shutdown all OBS workers if Lars hit the “I'm bored” button, …)

```
#!/bin/bash
if [ -z "$5" ]; then
    echo "Called with wrong number of arguments" >&2
    exit 1
fi
case "$1" in
  CRITICAL)
    case "$2" in
      SOFT)
        case "$3" in
            3)
                ssh -i /etc/nagios/keys/$4 root@$4 "/etc/init.d/$5 restart"
            ;;
        esac
      ;;
      *)
        # Looks like a HARD state, inform Admin via Nagios
      ;;
    esac
  ;;
  *)
    # OK, nothing to do
  ;;
esac
```

Active vs. passive monitoring

Active monitoring
Monitoring server actively checks the host or service
• Higher load on the monitoring server (SSH, xinetd, nrpe, ...) • Monitoring server needs access to the monitored machine • DoS => monitored machine ? • Allows “remote view” on external services

SNMP – old, but still useful

- SNMPv3 is more secure than NRPE 2.x (not 3.x)
- Use extend to execute local scripts

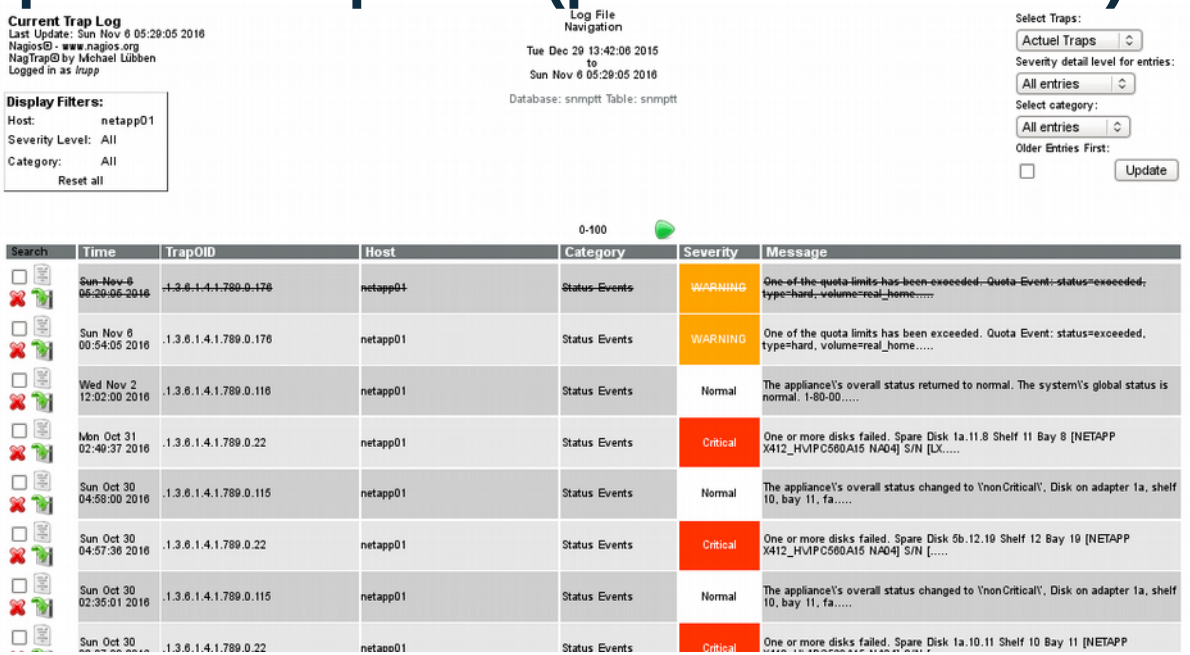
```
extend    test1    /bin/echo  "Hello, world!"
```

  snmpwalk -v2c -c public localhost nsExtendOutput1

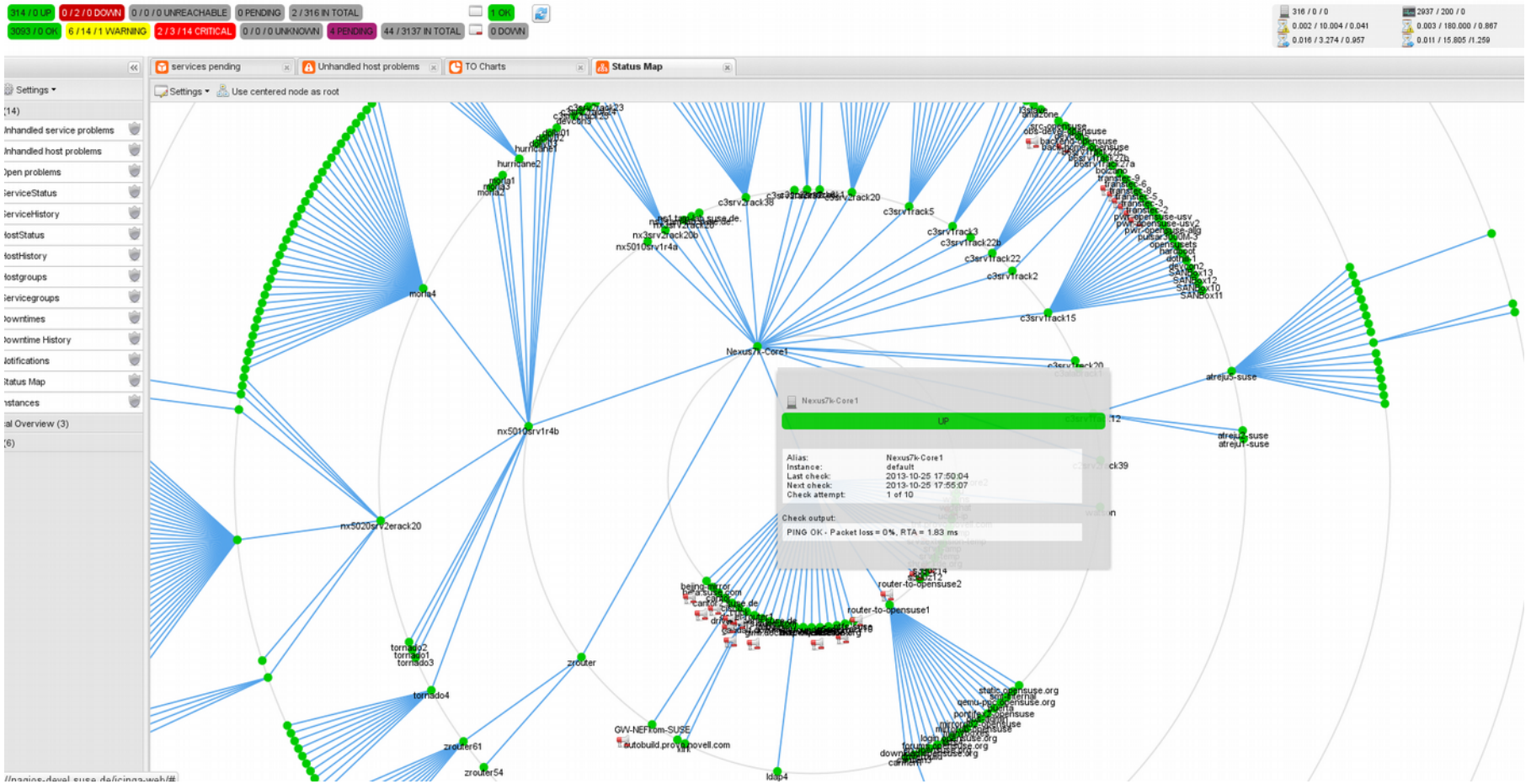
- Want to know which packages are installed ?

  snmpwalk -v2c -c public localhost hrSWInstalledName

- SNMP traps vs. snmpwalk (passive vs. active)

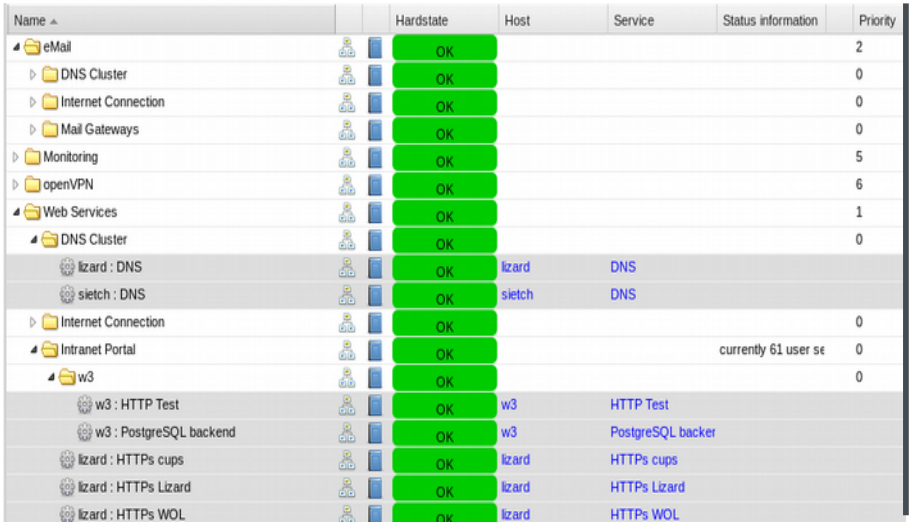


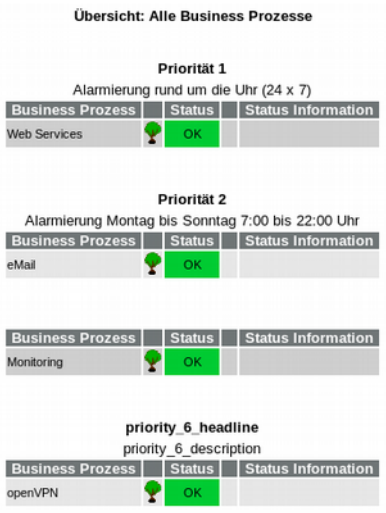
Why you always should define dependencies



What should be monitored?

Administrator View	Business View
Hardware health	Service health
Service availability – host based	Service availability – business based
Overview about the services and incidents of single hosts	Overview about the final business impact, not the service components
Only important for Administrators	Important for Managers and Customers





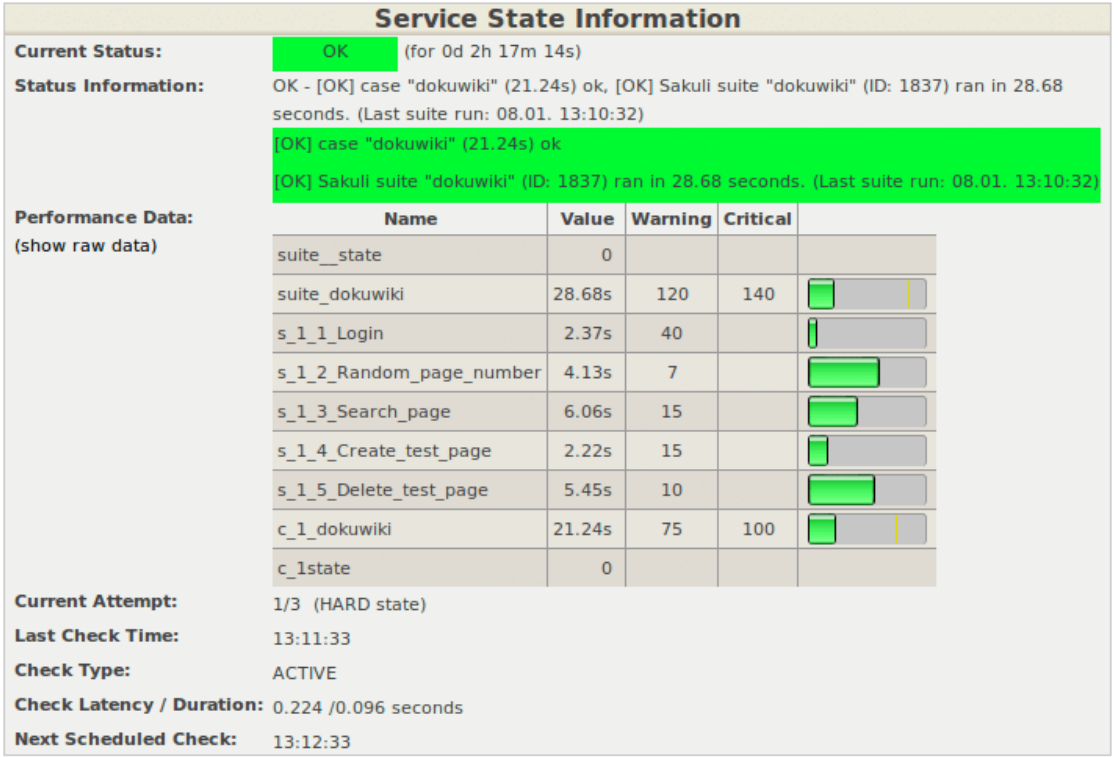
Hosts: what should be checked?

- “movable equipment” like FANs, hard drives, etc. are a must have
  (via BMC, IPMI, sensors, smart, ...)
- RAM usage – and ECC errors! (→ mcelog)
- CPU load, disk fill rate, network bandwidth – the “standard”
- Your services – from a customer view point

Applications: Check the user view!

- If possible, ask the developers how they test their software – and use their tools!
- Think about openQA, if you have Linux in use
- Everything is better than a simple “process is running”

- Below is an example from https://github.com/ConSol/sakuli
  Sakuli: Sahi (automation and testing tool for web applications)
  + Sikuli (image recognition to identify and control GUI components)



Service State Information

Current Status:	OK (for 0d 2h 17m 14s)
Status Information:	OK - [OK] case "dokuwiki" (21.24s) ok, [OK] Sakuli suite "dokuwiki" (ID: 1837) ran in 28.68 seconds. (Last suite run: 08.01. 13:10:32)
	[OK] case "dokuwiki" (21.24s) ok
	[OK] Sakuli suite "dokuwiki" (ID: 1837) ran in 28.68 seconds. (Last suite run: 08.01. 13:10:32)

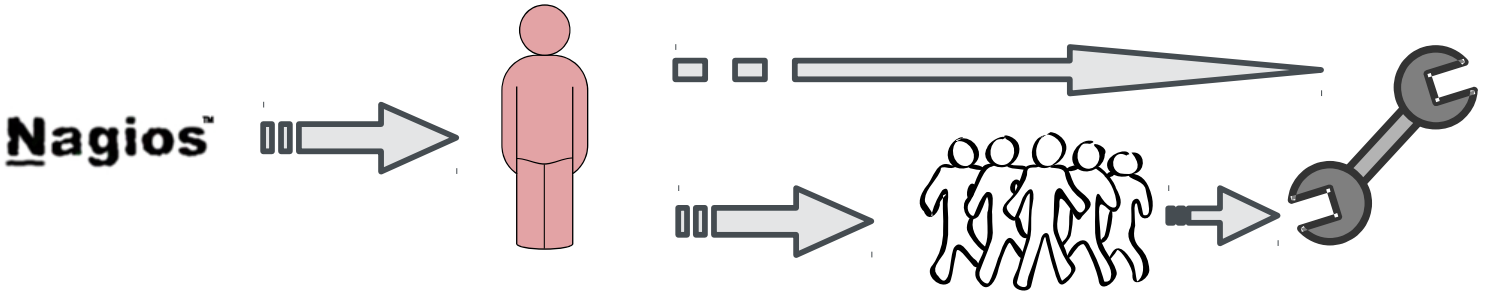
Performance Data: (show raw data)

Name	Value	Warning	Critical
suite__state	0		
suite_dokuwiki	28.68s	120	140
s_1_1_Login	2.37s	40	
s_1_2_Random_page_number	4.13s	7	
s_1_3_Search_page	6.06s	15	
s_1_4_Create_test_page	2.22s	15	
s_1_5_Delete_test_page	5.45s	10	
c_1_dokuwiki	21.24s	75	100
c_1state	0		

Current Attempt:	1/3 (HARD state)
Last Check Time:	13:11:33
Check Type:	ACTIVE
Check Latency / Duration:	0.224 /0.096 seconds
Next Scheduled Check:	13:12:33

Notifications → Escalations

- **Responsibility Groups = Notification Groups**
- **SMS notification for group leaders, if wanted**
- **Using escalations => reduce noise for Team members**
- **Usage of time frames:**
  - NO mail during non-work hours, including vacation
  - NO SMS during work hours and vacation
- **Bot: sends notifications to IRC during work hours**



Example scripts

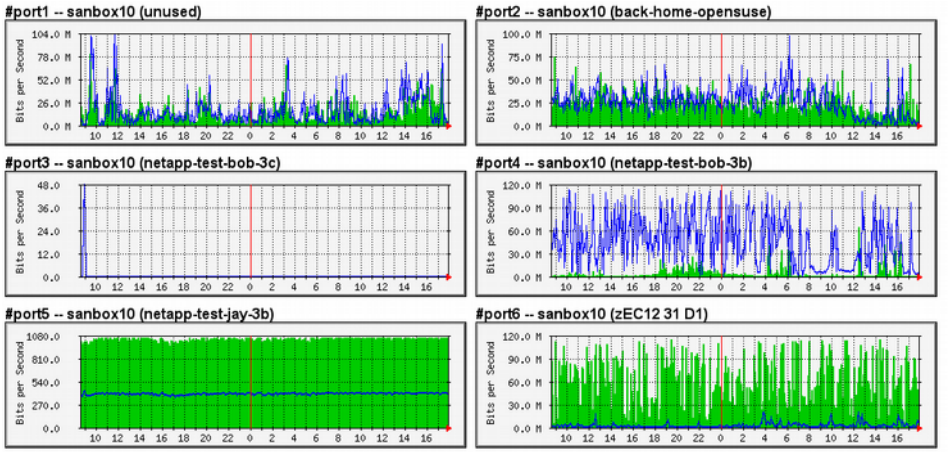
Monitoring SANBoxes with MRTG

For Qlogic, run the following command on your MRTG machine:

```
/usr/bin/cfgmaker --global "WorkDir: /srv/www/htdocs/mrtg"
--global "Options[_]: growright, bits, unknaszero"
--ifdesc=alias,name --ifref=name --noreversedns --no-down
--show-op-down --subdirs=sanbox-1 –output=/etc/mrtg/sanbox-
1.conf --snmp-options=:::::2 192.168.0.1
```

...or for Cisco MD:

```
/usr/bin/cfgmaker --global "WorkDir: /srv/www/htdocs/mrtg"
--global "Options[_]: growright, bits, unknaszero"
--ifdesc=alias --noreversedns --no-down --show-op-down –
subdirs=sanbox-2 –output=sanbox-2.conf --snmp-options=:::::2
192.168.0.2
```



Monitoring IO on your machines

On the machine your want to monitor:

- **Install monitoring-plugins-sar-perf**
- **Prepare a command like (NRPE example):**

```
command[check_iostat_home]=/usr/lib/nagios/plugins/check_iostat -d
root-fs_home -w 120000,120000,120000 -c 150000,150000,150000 -W 30
-C 50
```

Maybe also enable sysstat (systemctl enable sysstat), to have the data available on the host directly

MRTG graphs for network interfaces of virtual machines

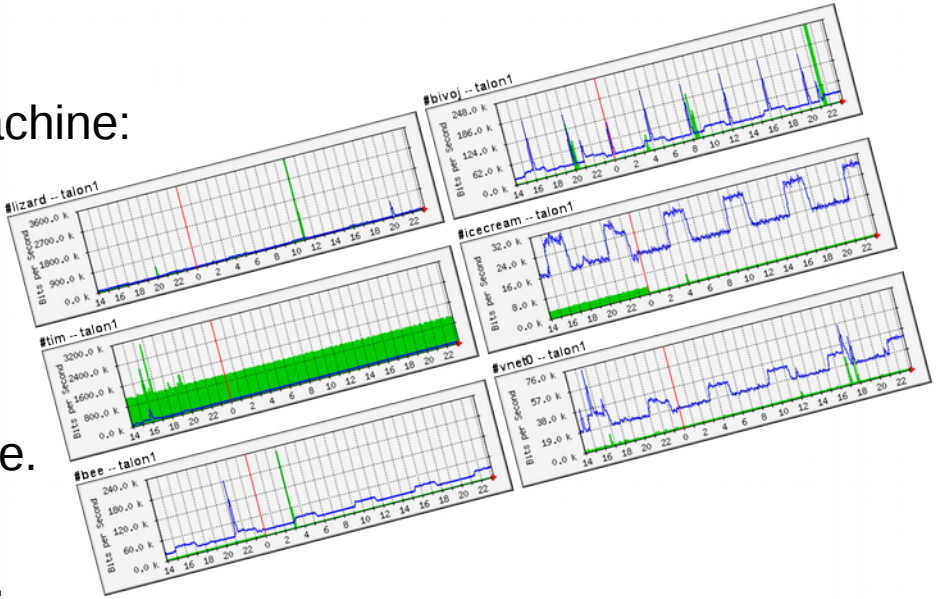
On the Server running the virtual machines, edit `/etc/snmp/snmpd.conf` :

```
[...]
rocommunity public 10.0.0.0/16
[...]
```

...and edit the xml definition of your virtual machine:

```
<interface type='bridge'>
  [...]
  <target dev='vm1'/>
  [...]
</interface>
```

Now (re-)start snmpd and your virtual machine.



On your MRTG machine, run:

```
/usr/bin/cfgmaker --global "WorkDir:
/srv/www/htdocs/mrtg" --global "Options[_]:
growright, bits, unknaszero" --ifdesc=alias,name
--ifref=name --noreversedns --no-down --show-op-down
--subdirs=vmserv1 --output=vmserv1.conf --snmp-
options=:::::2 10.0.0.101
```

Monitoring of MySQL servers

We are currently using two different checks:

`check_mysql` (monitoring-plugins-mysql package)

`check_mysql_health` (monitoring-plugins-mysql_health package)

You need a database user with "SELECT" access for both plugins. Usually, this means that you create a user named "nagios" or “monitor” in MySQL:

```
mysql> GRANT SELECT on nagios.* TO 'nagios'@'localhost' IDENTIFIED
BY 'nag1os';
mysql> flush privileges;
mysql> quit
```

Afterward you should be able to check the database via:

```
/usr/lib/nagios/plugins/check_mysql -H $HOST -u §USER -p $PASS
```

or:

```
/usr/lib/nagios/plugins/check_mysql_health --units MB --mode \
  threads-connected --username $USER --password $PASS \
  --warning 40 --critical 50
```

Monitoring of PostgreSQL

check the file `pg_hba.conf` on the database server to contain the correct IP addresses of the monitoring cluster

create the monitor user via the createuser command as user postgres:

```
postgres@pg1:~> createuser --pwprompt --interactive monitor
Enter password for new role:
Enter it again:
Shall the new role be a superuser? (y/n) y
Shall the new role be allowed to create databases? (y/n) n
Shall the new role be allowed to create more new roles? (y/n) n
```

Note: the SUPERUSER privilege is needed for some special checks like "archive_ready" – you might want to skip this.

restart the database

Try on the monitoring cluster:

```
~> ./check_postgres.pl --dbpass=$PASSWORD -dbuser=$USERNAME \
--action=archive_ready -H pg1
POSTGRES_ARCHIVE_READY OK: DB "postgres" (host:pg1) WAL ".ready" files
found: 0 | time=0.02s files=0;10;15
```

...and there is more...

More and more monitoring-plugins* packages come with enabled Apparmor profiles: check /var/log/audit/audit.log if something seems to be crazy

Re-enable notifications automatically via cron – to not forget it:

```
#!/bin/bash

CFG=/etc/icinga/icinga.cfg
commandfile=$(grep ^command_file "$CFG" | awk -F'=' '{ print $2 }')
if [ -p "$commandfile" ]; then
        now=`date +%s`
        printf "[%lu] ENABLE_NOTIFICATIONS\n" $now > "$commandfile"
fi
```

Monitor your NSCA daemon via `monitoring-plugins-nsca` and a dummy test (see README)

Create performance data for your monitoring:

```
#!/bin/bash
if /etc/init.d/icinga status >/dev/null 2>/dev/null ; then
 if [ -p /var/run/icinga/icinga.cmd ]; then
  su - icinga -c "/usr/lib/nagios/plugins/check_nagiostats\
     --EXEC /usr/sbin/icingastats --passive $HOST \
     icingastats  >> /var/run/icinga/icinga.cmd"
 fi
fi
```

Monitor your monitoring setup!

More package recommendations

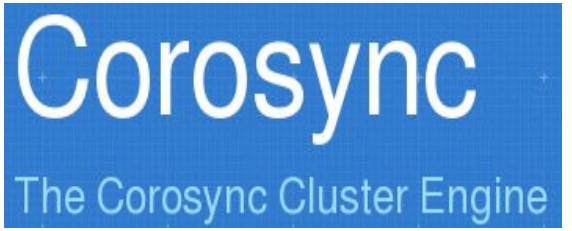
- monitoring-plugins-zypper :
  - check for security, recommended or optional updates, list affected packages
  - warns, if an installed package is not from the official channel or whitelisted
- monitoring-plugins-nsca :
  - checks, if your NSCA daemon is able to submit data to your monitoring instance
- monitoring-plugins-bind9 :
  - interesting DNS statistics …
- monitoring-plugins-mailstat :
  - more interesting Mail server statistics
- check_ssl_cert :
  - check your own SSL certificates
- check_mk-agent* :
  - collect mass amount of data from your operating system
- monitoring-plugins-sap_health :
  - check various parameters of a SAP system

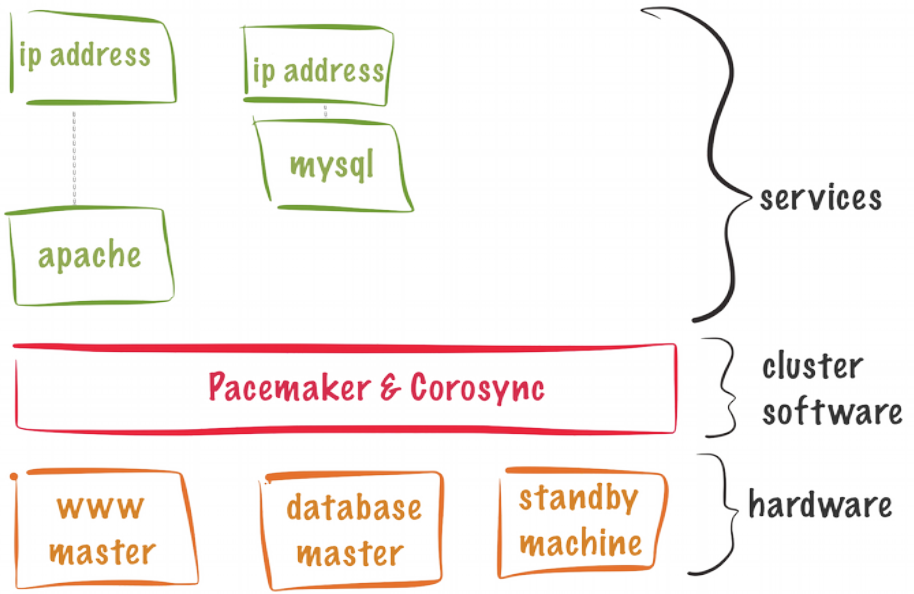
Demo time?

High available monitoring

High Availability

(requires SUSE Linux Enterprise High Availability Extension)





Basic rules

Services implementing HA on their own:

- Prefer the integrated solution
- For example MySQL, DHCP, named (bind), PostgreSQL, ...

Services can run independent on the node:

- Keep running independent (but monitor) or run in clone mode
- For example ido2db, NSCA, gearmand, apache, nrpe, ...

You can run more then one DRBD resource via Pacemaker:

- Helps to run on different storage (SAN vs. Harddisk vs. SSD)
- Helps with load balancing (use different storages for different tasks)

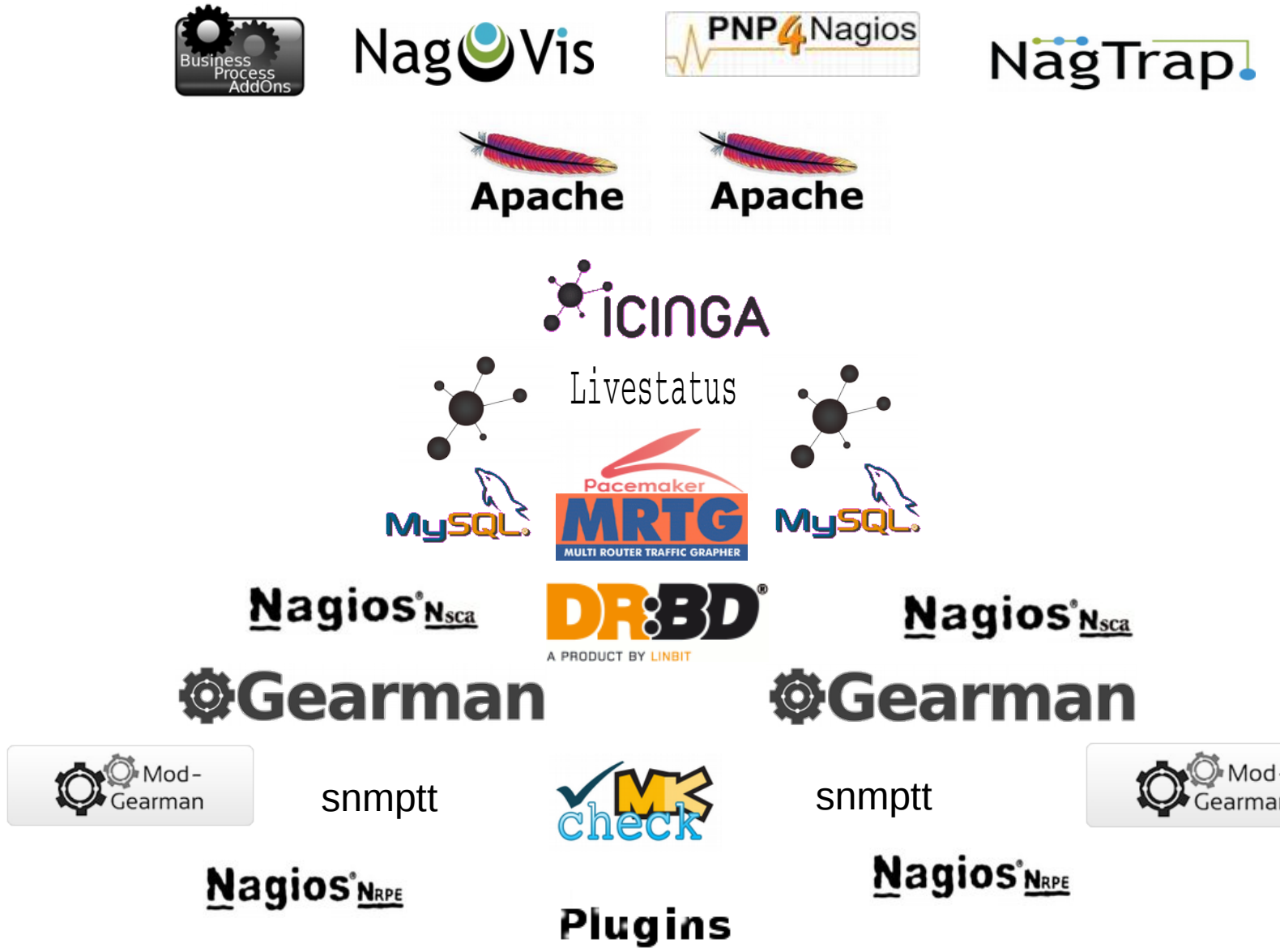
Have a third node at least for Quorum

- This allows corosync to decide which host is “right” in a split brain situation
- The 3rd node might be a simple virtual machine just joining for quorum

Basic overview of the demo setup

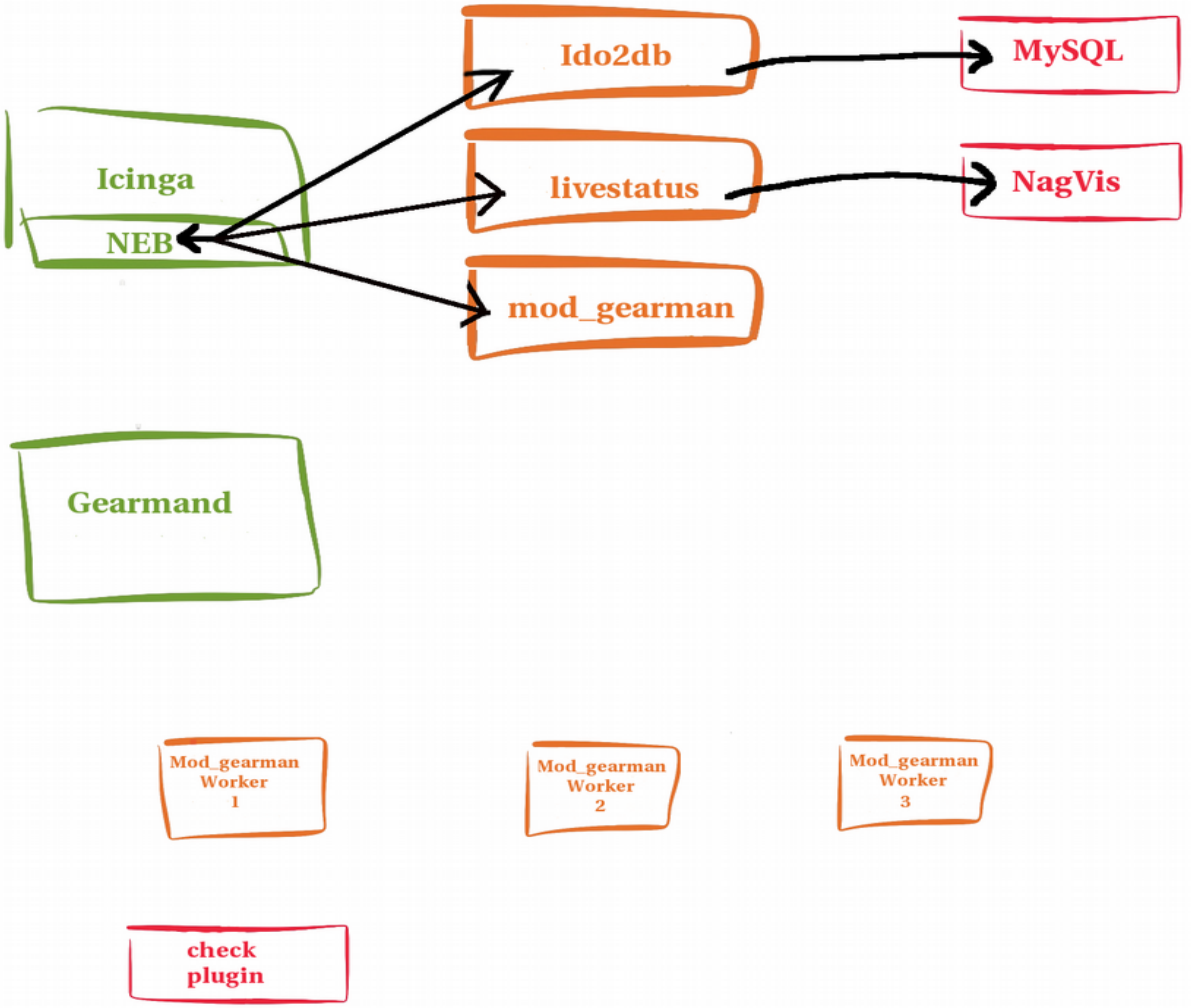
- **Corosync/Pacemaker** Cluster (two main machines + one VM just for quorum) – using IPMI for STONITH
- **DRBD** to provide storage (PNP, Logs) on both machines
- Services like MySQL (cluster), snmptrapd or NSCA run “**unmanaged**” on all nodes
- **mod_gearman** for Load-Balancing of normal checks
- **check_mk** for automatic checks and Load-Reducing
- **MRTG** for statistics from Network and SAN (for historical reasons)
- Lot's of (web) add-ons for different tasks (NagVis, PNP, NagiosBP, ...)

Load-Balanced / HA Monitoring in project pictures

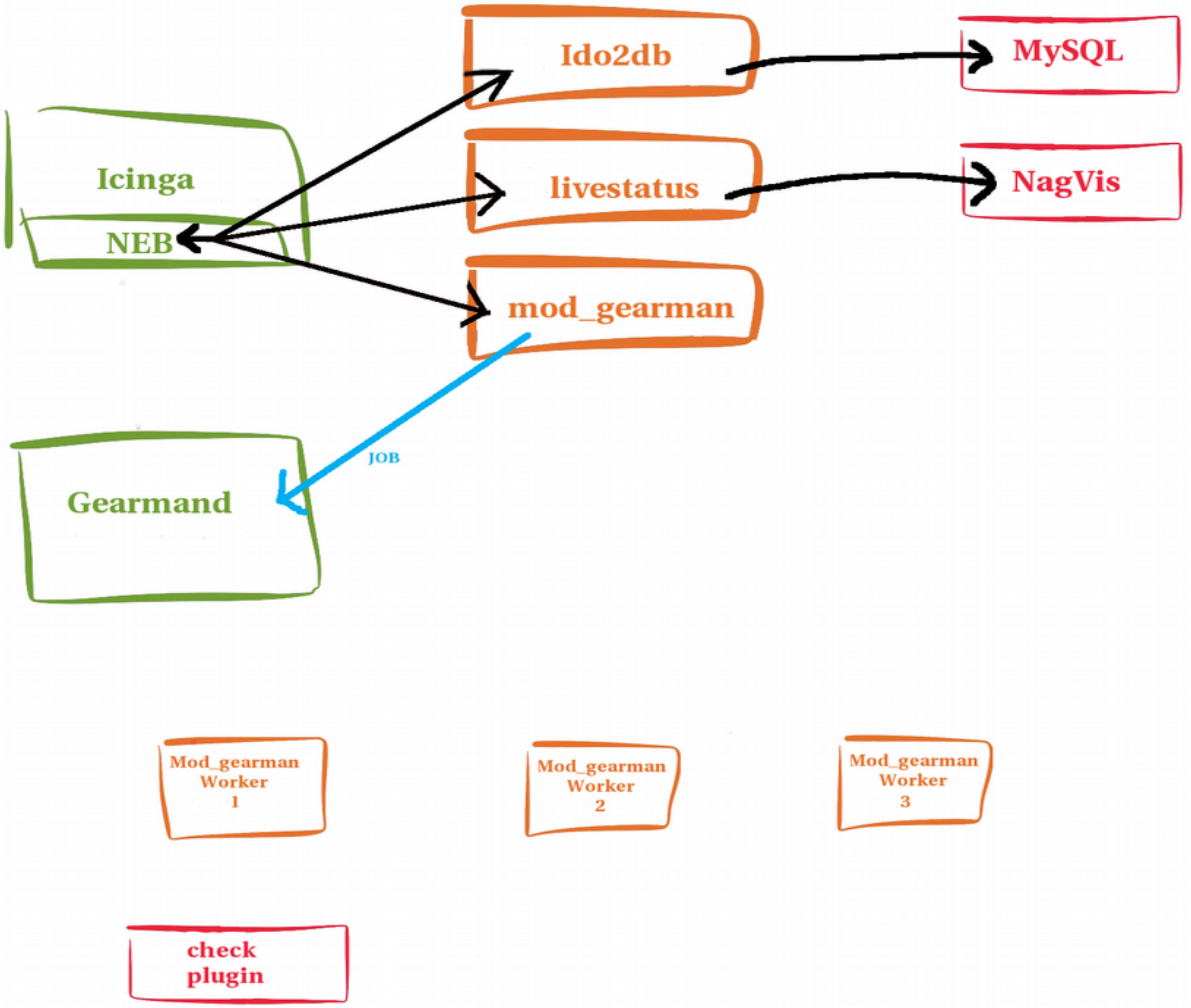


Load balanced monitoring

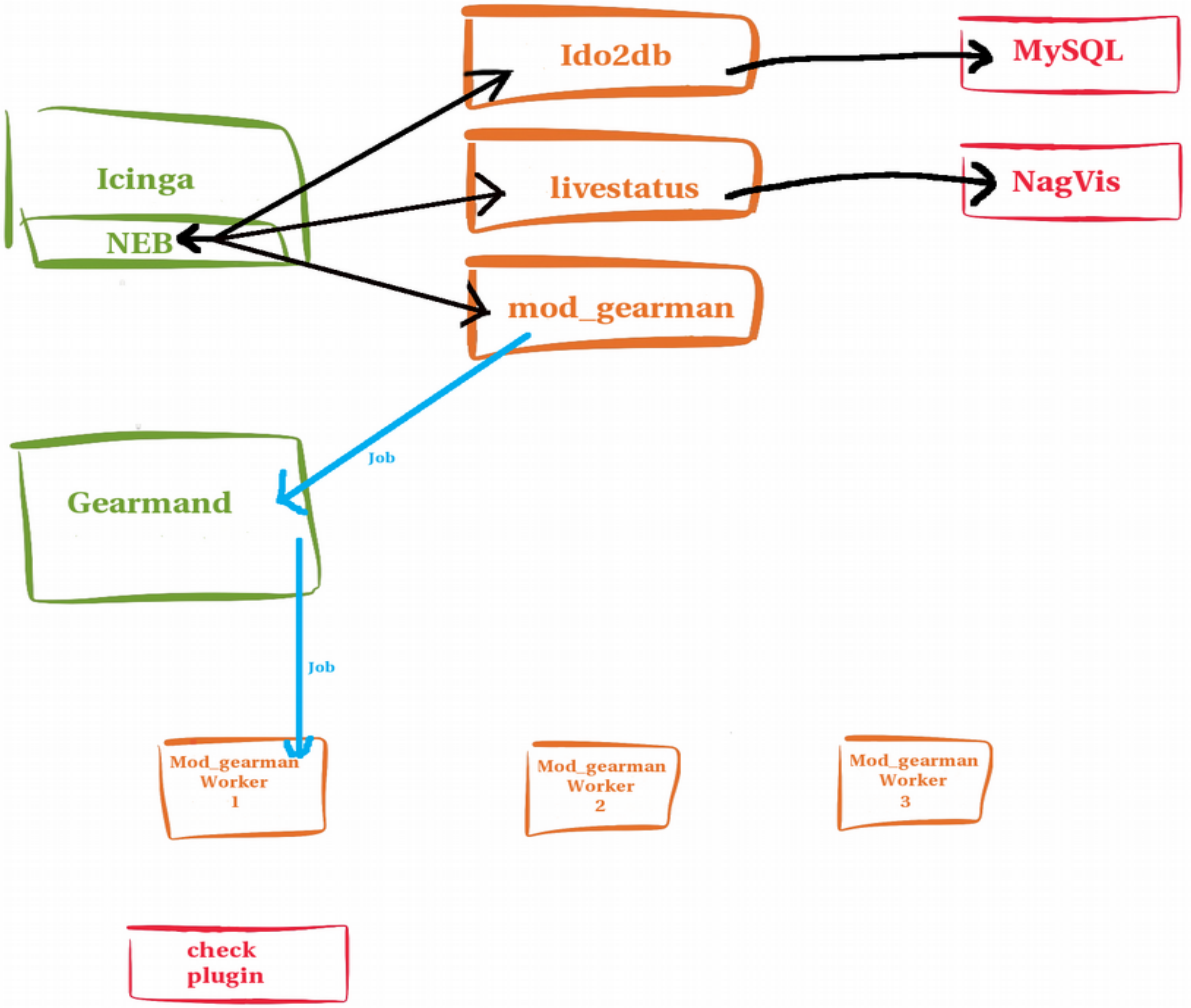
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Gearmand
Mod_gearman Worker 1
Mod_gearman Worker 2
Mod_gearman Worker 3
check plugin

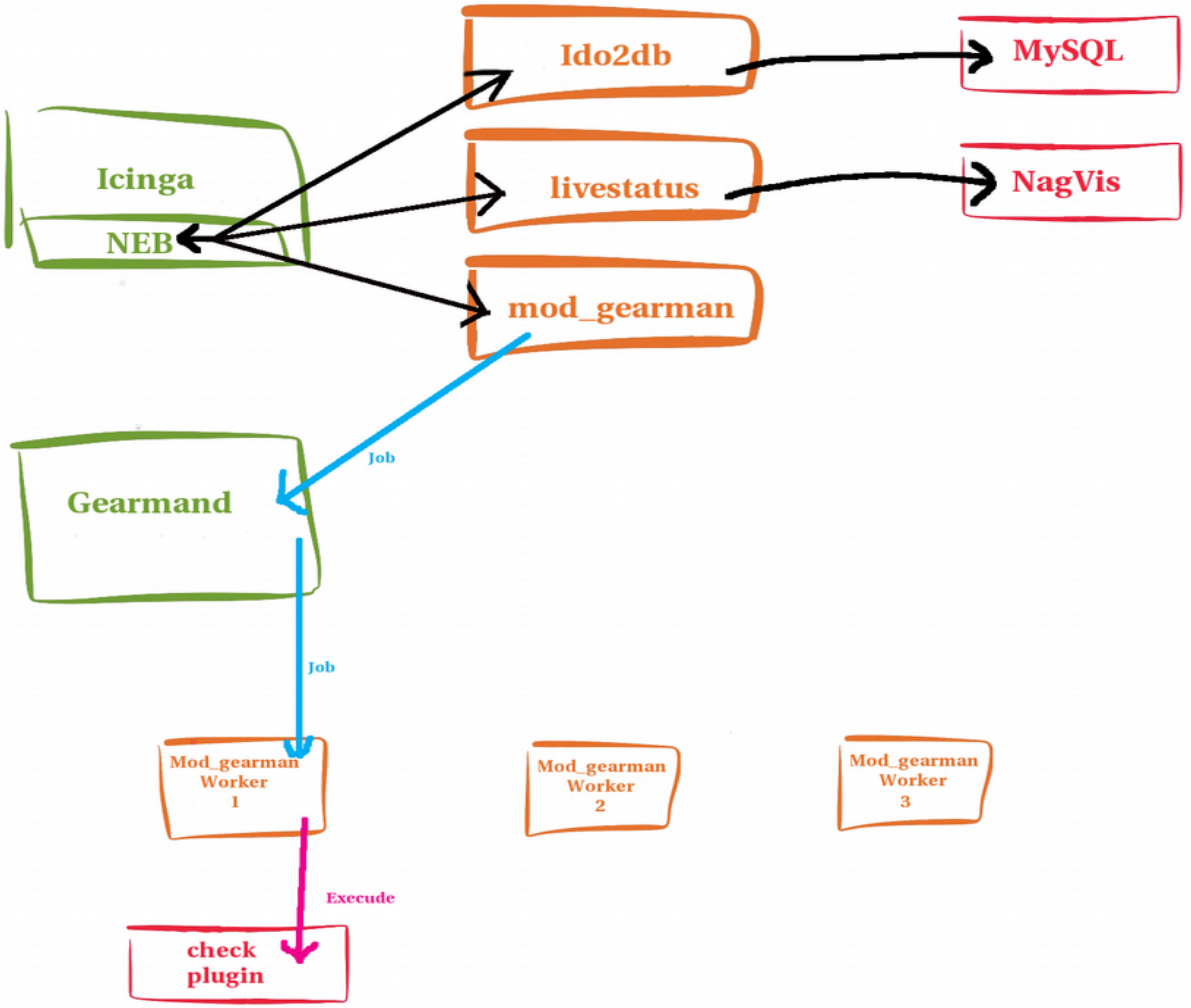
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
JOB
Gearmand
Mod_gearman Worker 1
Mod_gearman Worker 2
Mod_gearman Worker 3
check plugin

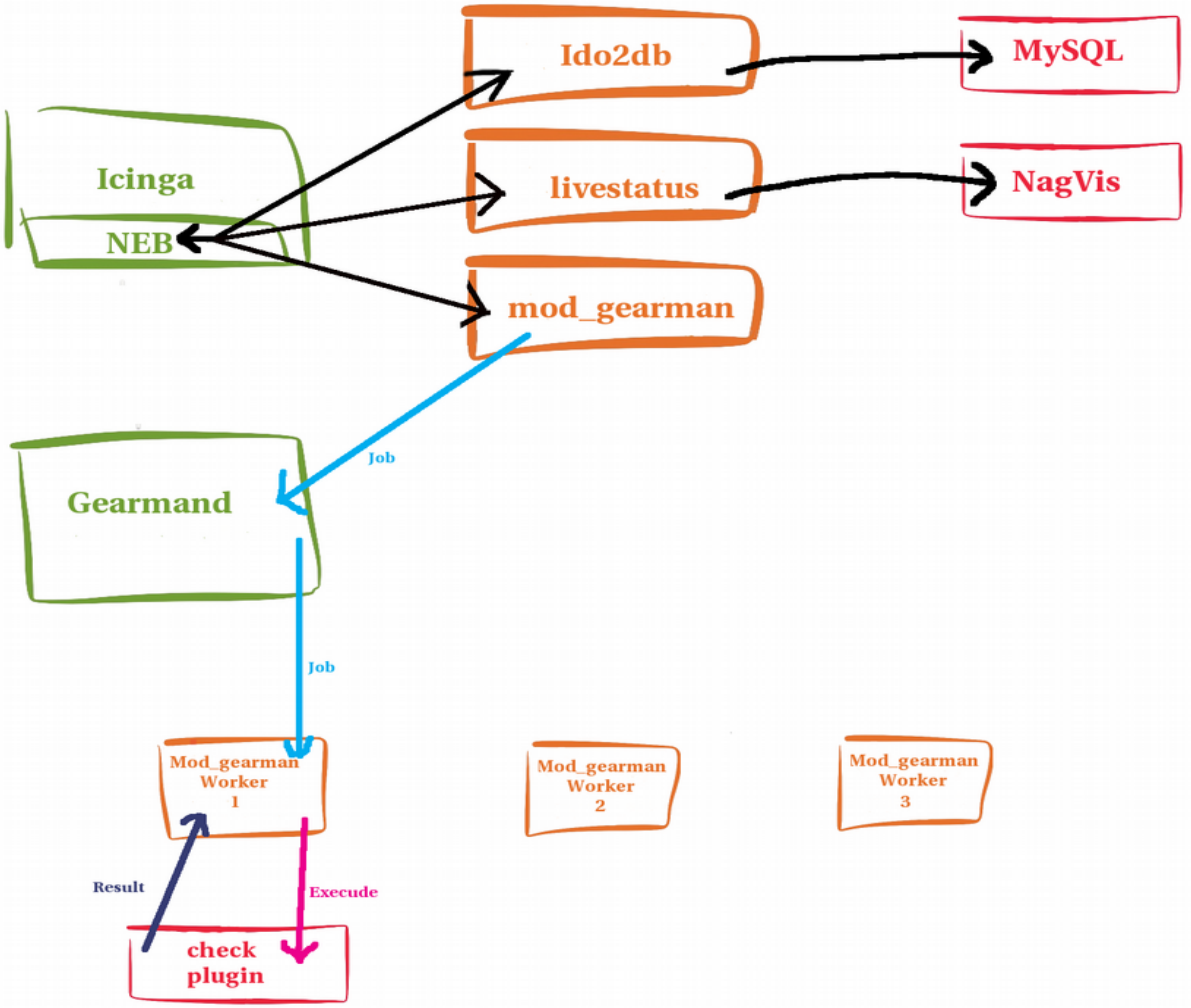
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Job
Gearmand
Job
Mod_gearman
Worker
1
Mod_gearman
Worker
2
Mod_gearman
Worker
3
check
plugin

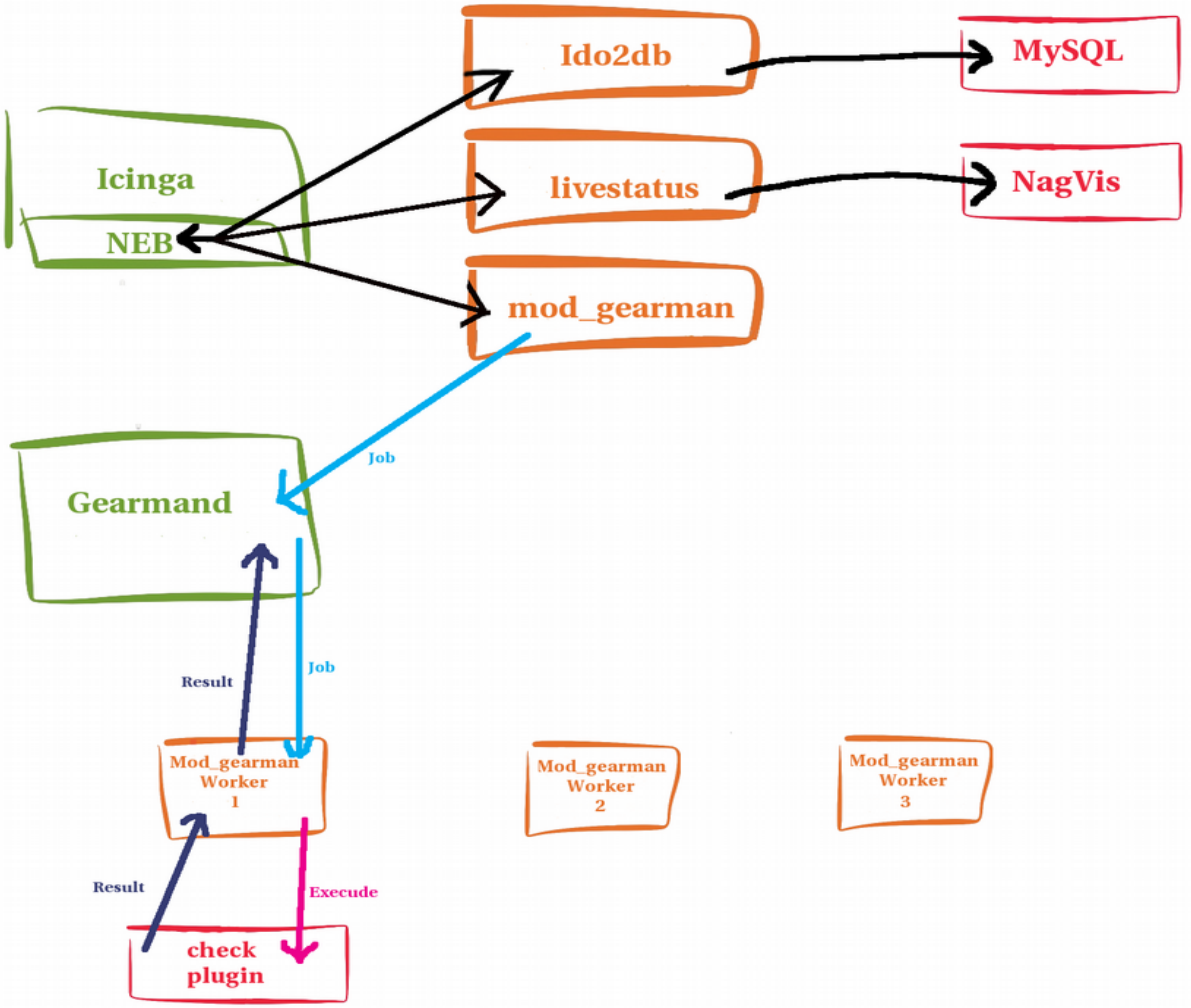
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Job
Gearmand
Job
Mod_gearman
Worker
1
Mod_gearman
Worker
2
Mod_gearman
Worker
3
Execude
check
plugin

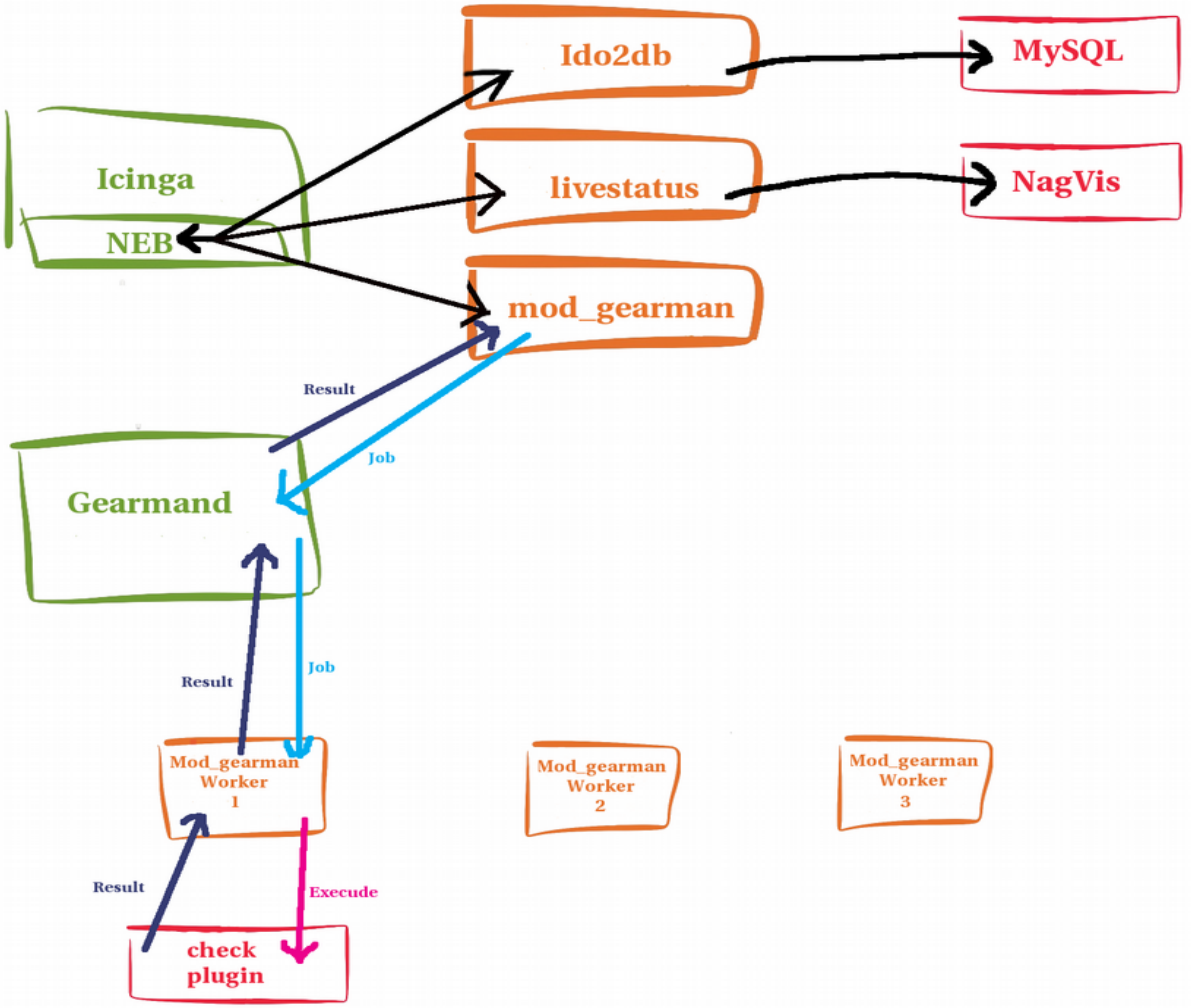
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Job
Gearmand
Job
Mod_gearman
Worker
1
Mod_gearman
Worker
2
Mod_gearman
Worker
3
Result
Execude
check
plugin

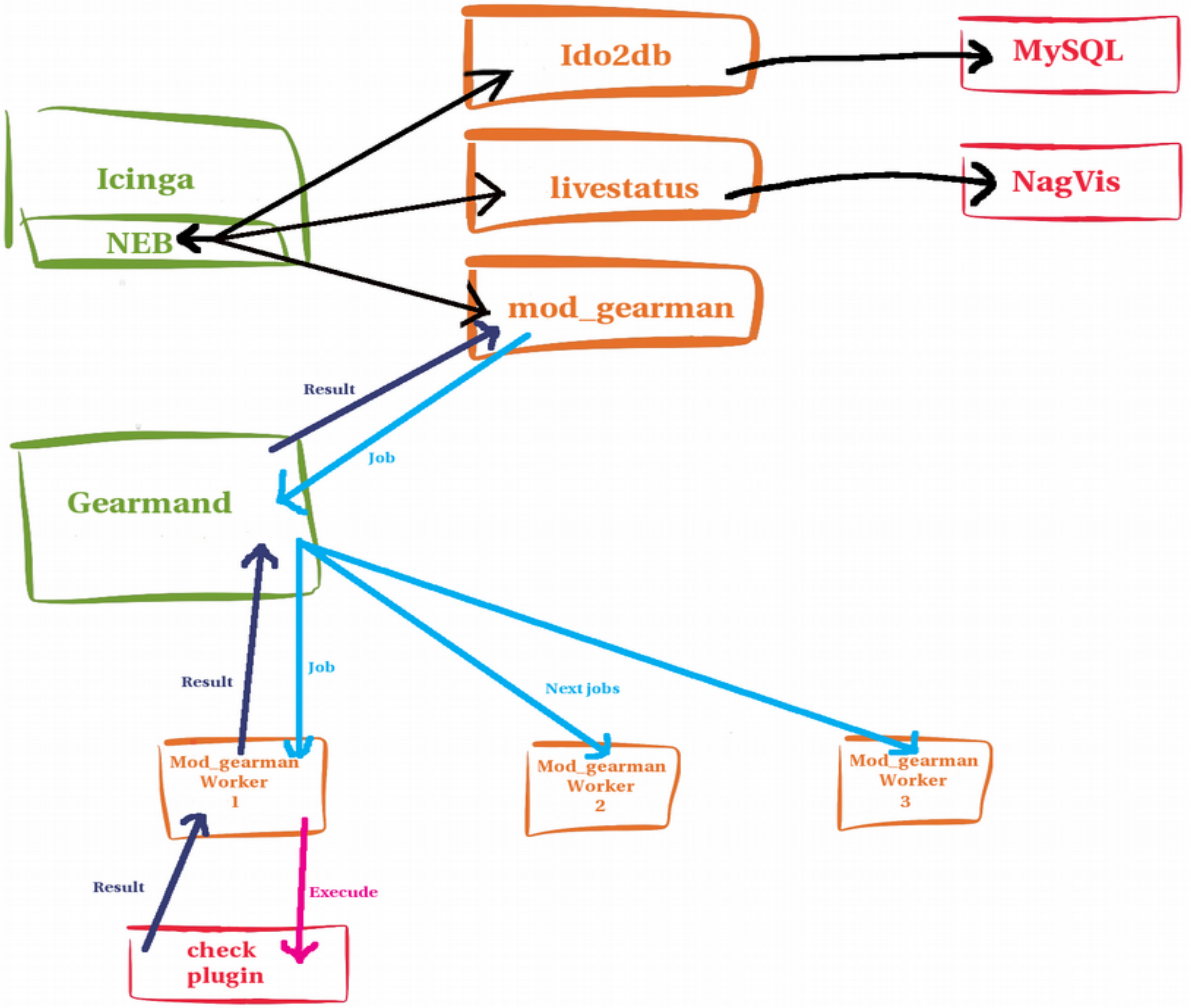
Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Job
Gearmand
Result
Job
Mod_gearman Worker 1
Mod_gearman Worker 2
Mod_gearman Worker 3
Result
Execude
check plugin

Gearman

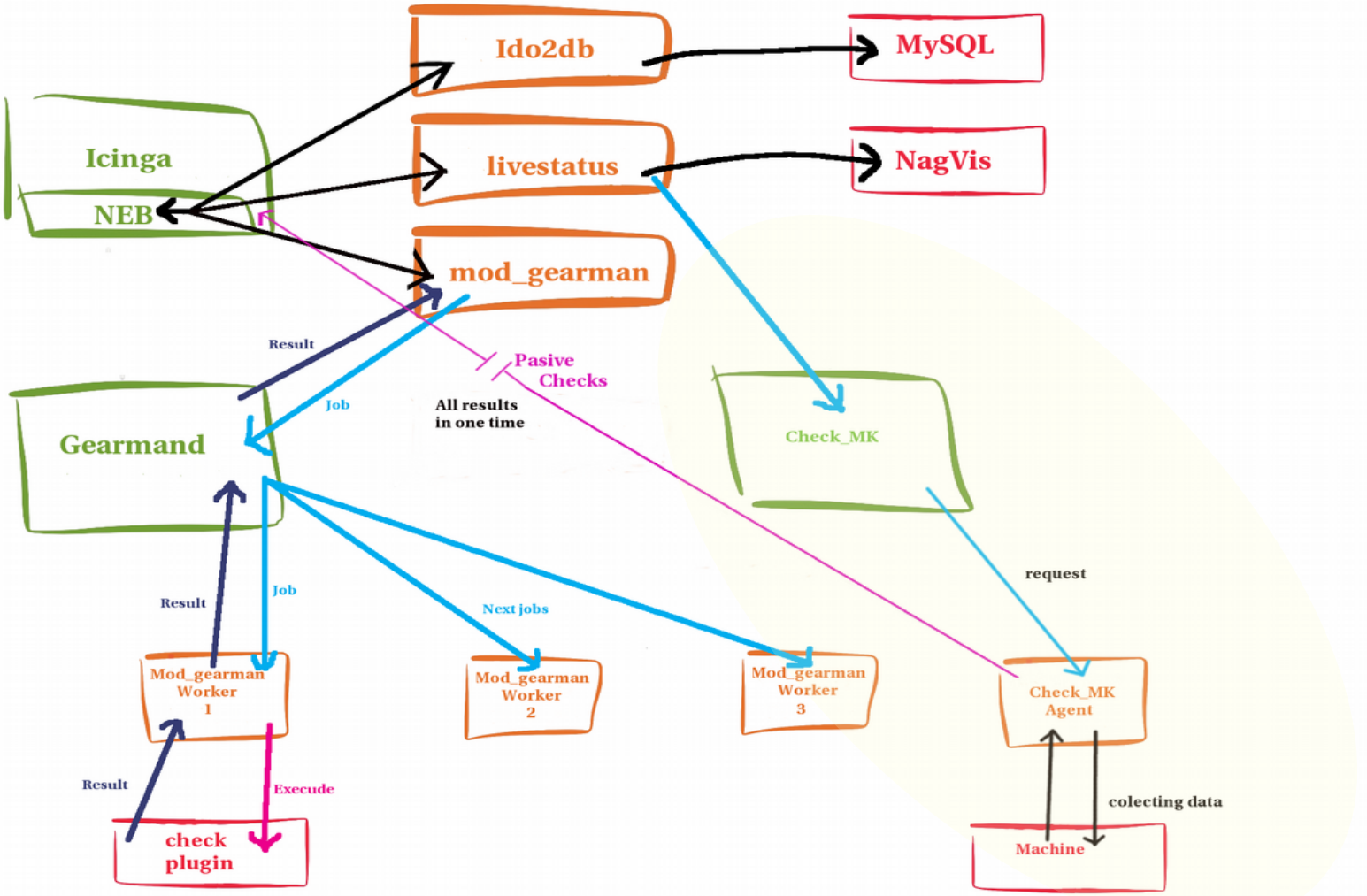

Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Result
Job
Gearmand
Result
Job
Mod_gearman
Worker
1
Mod_gearman
Worker
2
Mod_gearman
Worker
3
Result
Execude
check
plugin

Gearman


Ido2db
MySQL
Icinga
livestatus
NagVis
NEB
mod_gearman
Result
Job
Gearmand
Result
Job
Next jobs
Mod_gearman Worker 1
Mod_gearman Worker 2
Mod_gearman Worker 3
Result
Execude
check plugin

Monitoring via check_mk

Check MK


Ido2db
MySQL
Icinga
NEB
livestatus
NagVis
mod_gearman
Result
Pasive
Checks
Job
All results
in one time
Gearmand
Check_MK
request
Result
Job
Next jobs
Mod_gearman
Worker
1
Mod_gearman
Worker
2
Mod_gearman
Worker
3
Check_MK
Agent
Result
Execude
colecting data
check
plugin
Machine

Demo time?

Entering: Salt

Using Salt for Monitoring?

- ZeroMQ provides a secured communication channel (other than NRPE 2.x for example)
- Messages reach all or one or a group of clients at the “same time” - easily
- Clients can sent the results to the bus and forget about it
- The “Event System” allows to execute “stuff” on demand
- Reducing the number of running services is always a good thing
- The Salt Scheduler is able to schedule checks at different times with less configuration overhead

Demo time?

Monitoring Future @SUSE

Monitoring future @SUSE

Nothing below is decided yet. Feel free to tell me your opinion either directly or via Email to Lars.Vogdt@suse.com - Thanks!

- Will there be a successor of Icinga (like Icinga 2, Sensu or Zabbix or Naemon)? **Your** opinions, please
- Obsole NRPE / NSCA in favor of Salt
- Consolidate user and group used for monitoring - current favorite is monitor:monitor
- Provide more monitoring plugins in Package Hub
- Provide either some “best practices” guidelines or a “monitoring appliance” - maybe both ?

Links and other information

Links

https://en.opensuse.org/Special:Search/all:Nagios~

http://docs.icinga.org/latest/en/

https://www.suse.com/support/update/announcement/2015/suse-ou-20151252-1.html

http://mathias-kettner.com/check_mk.html

Other sessions

Thursday, Nov 10, 4:30 PM - 5:30 PM:

- BOV89296 - SUSE Best Practices - Sharing Expertise, Experience and Knowledge

Friday, Nov 11, 9:00 AM - 10:00 AM

- FUT92726 - The SUSE Manager Roadmap: A journey towards agile management of workloads in the enterprise

Friday, Nov 11, 10:15 AM - 11:15 AM

- FUT95338 - SUSE Package Hub - Community Packages for Enterprise Users

Best Practices in Monitoring

Lars Vogdt
Team Lead SUSE DevOPS
<Lars.Vogdt@suse.com>

- Server outages carry a potential impact to business continuity, better check more than less on your machines – often hardware causes a problem in software...

- Avoid false positives
- the right problem at the right time
- Identify possible bottle necks or single point of failures early

- ITIL draws your focus to the business/service side
- Summarize monitoring data for the appropriate audience
- => Executive stakeholders
- => Team Leads
- => Frontline of maintenance teams
- Know your Audience, and what gets their attention