How we learned to stop worrying and love Zabbix

Nagios->Zabbix migration
Some background info on our monitoring ecosystem
How we evolved:

5 companies, < 100 hardware items

Various tools (FPinger, scripts)

30 companies, 10 engineers, <=1000 hardware items

Linux/Windows/Solaris/*BSD/Cisco/…/…

Nagios (NCSA/NRPE/NagiosGraph/NagVis) and MOM

100+ companies, 20+ engineers, >=5000 hardware items

Linux/Windows/Solaris/*BSD/Cisco/VoIP/Security

Centreon
Why Centreon?

• ACL

• Distributed monitoring

• Nagios backend and plugins

• Not so ugly

• Web configuration
Things to monitor

- *nix and Windows (50-100+ sensors per host)
- VoIP infrastructure (H/W, S/W, trunks and calls)
- Databases and applications
- Network equipment (interfaces, connectivity etc.)
- Supplementary systems (cameras, temperature sensors, UPSes etc.)
How it worked

Monitoring automation

- NEW HOST
- PUPPET (ROLES AND SERVICES)
- HOST CONFIGURATION EXPORT FROM HOST TO PUPPET
- CENTREON CONFIGURATION EXPORT TO CENTREON
- HOST IS MONITORED

Server infrastructure
Problems

- Management
- Not enough information and features
- Performance problems
  - NCSA/NRDP limitations
  - need to use a lot of active checks
  - script performance
- Basic agents (NRPE/NCPA)
- WMI check performance
- Nagios.cfg limitations
- Broker management issues (SSH+scripts)
- Not enough SLA
- Too many different services to integrate together (brokers, processors, scripts, etc)
Monitoring management problems
Plain nagios architecture
Extended nagios architecture (Nagios XI)
My case (Centreon)
+CLAPI
+Nagvis
+Puppet
+home-made scripts
+hacks
+many more
Synchronization issues
Performance data shipping issues
Any configuration change => config rebuild + engine reload
Data is distributed across different files and databases
Interaction is based on sockets, files and scripts
Automation problems in action
Agents and checks
Nagios doesn’t decide if check result is ok or not. Script does

<table>
<thead>
<tr>
<th>check_nrpe_cpu</th>
<th>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_cpu -...</th>
<th>Check</th>
</tr>
</thead>
<tbody>
<tr>
<td>check_nrpe_cpu_proc</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_cpu_p...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_disk</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_disk ...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_diskio</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_diskio...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_load</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_load ...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_nginx_status</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_nginx...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_phys_mem</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_mem ...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_process</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_proc...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_uptime</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_uptime...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nrpe_users</td>
<td>$USER1$/check_nrpe -H $HOSTADDRESS$ -c check_users...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nt_cpu</td>
<td>$USER1$/check_nt -H $HOSTADDRESS$ -v CPULOAD -s &quot;p...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nt_disk</td>
<td>$USER1$/check_nt -H $HOSTADDRESS$ -v USEDDISKSPACE...</td>
<td>Check</td>
</tr>
<tr>
<td>check_nt_memuse</td>
<td>$USER1$/check_nt -H $HOSTADDRESS$ -v MEMUSE -s &quot;pu...</td>
<td>Check</td>
</tr>
</tbody>
</table>
Nagios doesn’t decide if check result is ok or not. Script does.

Nothing is built in

Only scripts

No proper authentication

No decent agents

History is tracked in per-check files
Performance

Exit codes

Interpreted languages (Perl, Python, bash, whatever..)
NOT ENOUGH SLA

CRITICAL
WARNING
OK
FLAPPING
UI
We decided to give Zabbix a try
We decided to give Zabbix a try
Less than a day to setup
Less than a day to setup a distributed installation
Less than a week to setup most of the sensors
It took less than two months to deploy it on 90% of the monitored infrastructures

- 1218 servers
- 519 network devices
- over 100k items
Migration process

Deploy: puppet + SQL-scripts
Checks: google and github

Built-in wmi.get!
TLS authentication!
Built-in inventory!
Built-in LLD!
JSON-API!
Lot more
How it works now

Monitoring automation

Server infrastructure

NEW HOST

AUTOMATED AGENT INSTALLATION

AUTOMATIC ZABBIX CONFIGURATION

HOST IS MONITORED

Database Server

Central Zabbix server

Remote proxy

Remote proxy

Remote proxy
Comparison

NEW HOST

AUTOMATED AGENT INSTALLATION

AUTOMATIC ZABBIX CONFIGURATION

HOST IS MONITORED

NEW HOST

PUPPET (ROLES AND SERVICES)

HOST CONFIGURATION EXPORT FROM HOST TO PUPPET

CENTREON CONFIGURATION EXPORT TO CENTREON

HOST IS MONITORED
What we like

• No more hassle adding new hosts
• Much more information
• Finally decent reports
• Better performance
• 3 times less resources required
Zabbix is friendly)
Questions?