4. Monitoring with Prometheus

Infrastructure Automation
HOGENT applied computer science
Bert Van Vreckem & Thomas Parmentier
2024-2025

Monitoring

Learning goals

- Understanding the concept of monitoring, observability
- Setting up a monitoring dashboard with Prometheus

What is monitoring?

Keep track of systems running in production

- Problem detection
- Troubleshooting
- Reporting and improvement

Part of *Reliability Engineering*

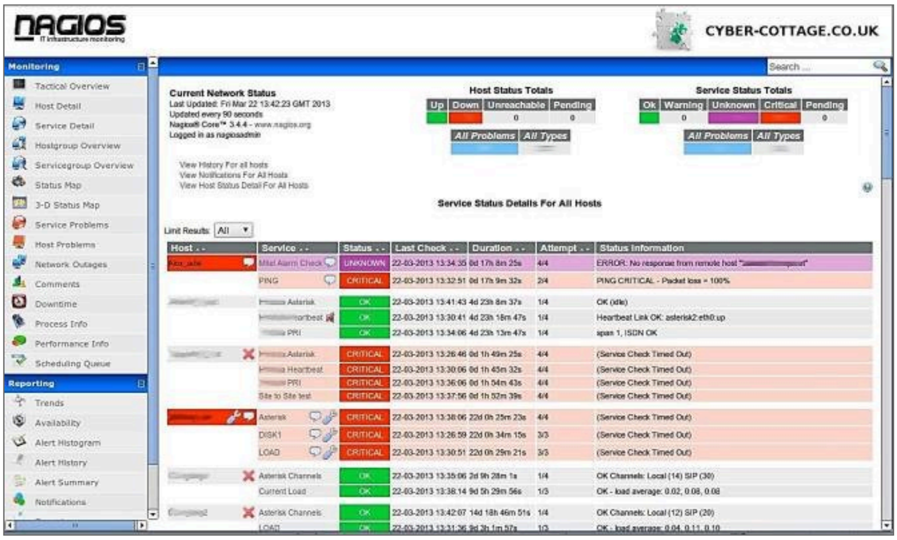
What to monitor?

Anything that matters for your business! -> **monitoring targets**

- Demand
- Workload
  - Availability, performance, faults/errors
- Resources (technical metrics)
  - e.g. CPU, connection pool capacity, …
- Business metrics

Early monitoring solutions: Alerting

- Nagios, Icinga, Zmon, Sensu, …
- Regularly execute checks (scripts)
- Nonzero exit status? Generate alert!



Current Network Status
Last Updated: Fri Mar 22 13:42:23 GMT 2013
Updated every 90 seconds
Nagios® Core™ 3.4.4 - www.nagios.org
Logged in as nagiosadmin

View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
🟩	🟥	0	0

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
🟩	0	🟪	🟥	0

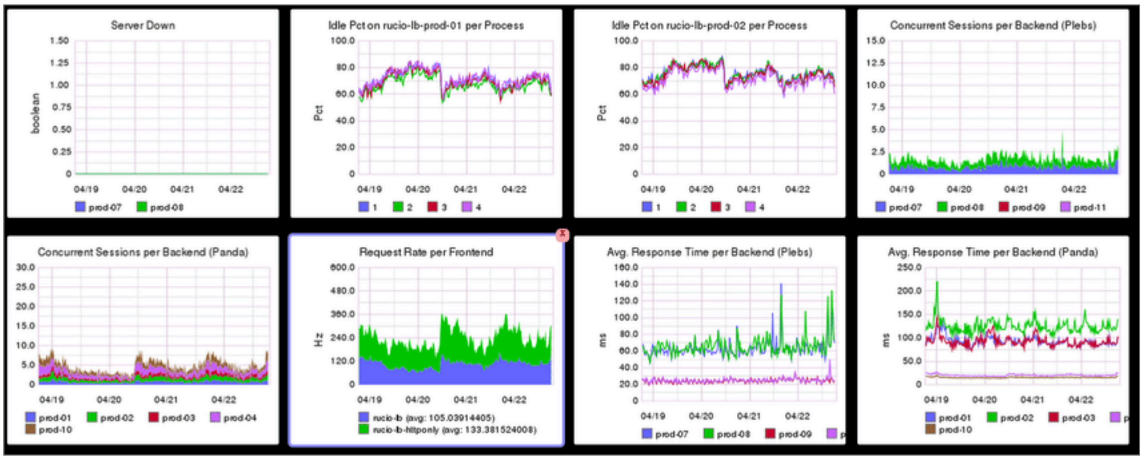
Service Status Details For All Hosts

Limit Results: All

Host	Service	Status	Last Check	Duration	Attempt	Status Information
[redacted]	Mitel Alarm Check	UNKNOWN	22-03-2013 13:34:35	0d 17h 8m 25s	4/4	ERROR: No response from remote host "[redacted]"
	PING	CRITICAL	22-03-2013 13:32:51	0d 17h 9m 32s	2/4	PING CRITICAL - Packet loss = 100%
[redacted]	[redacted] Asterisk	OK	22-03-2013 13:41:43	4d 23h 8m 37s	1/4	OK (idle)
	[redacted]ortbeat	OK	22-03-2013 13:30:41	4d 23h 18m 47s	1/4	Heartbeat Link OK: asterisk2:eth0:up
	[redacted] PRI	OK	22-03-2013 13:34:06	4d 23h 13m 47s	1/4	span 1, ISDN OK
[redacted]	[redacted] Asterisk	CRITICAL	22-03-2013 13:26:46	0d 1h 49m 25s	4/4	(Service Check Timed Out)
	[redacted] Heartbeat	CRITICAL	22-03-2013 13:30:06	0d 1h 45m 32s	4/4	(Service Check Timed Out)
	[redacted] PRI	CRITICAL	22-03-2013 13:36:06	0d 1h 54m 43s	4/4	(Service Check Timed Out)
	Site to Site test	CRITICAL	22-03-2013 13:37:56	0d 1h 52m 39s	4/4	(Service Check Timed Out)
[redacted]	Asterisk	CRITICAL	22-03-2013 13:38:06	22d 0h 25m 23s	4/4	(Service Check Timed Out)
	DISK1	CRITICAL	22-03-2013 13:26:59	22d 0h 34m 15s	3/3	(Service Check Timed Out)
	LOAD	CRITICAL	22-03-2013 13:30:51	22d 0h 29m 21s	3/3	(Service Check Timed Out)
[redacted]	Asterisk Channels	OK	22-03-2013 13:35:06	2d 9h 28m 1s	1/4	OK Channels: Local (14) SIP (30)
	Current Load	OK	22-03-2013 13:38:14	9d 5h 29m 56s	1/3	OK - load average: 0.02, 0.08, 0.08
[redacted]	Asterisk Channels	OK	22-03-2013 13:42:07	14d 18h 46m 51s	1/4	OK Channels: Local (12) SIP (20)
	LOAD	OK	22-03-2013 13:31:36	9d 3h 1m 57s	1/3	OK - load average: 0.04, 0.11, 0.10

Early monitoring solutions: Graphing

- E.g. Graphite, mrtg
- Backed by time series database filled with metrics
- Separate tool for data collection (e.g. collectd, statsd)


Server Down
boolean
1.50
1.25
1.00
0.75
0.50
0.25
0
04/19
04/20
04/21
04/22
prod-07
prod-08
Idle Pct on rucio-lb-prod-01 per Process
Pct
100.0
80.0
60.0
40.0
20.0
0
1
2
3
4
Idle Pct on rucio-lb-prod-02 per Process
Concurrent Sessions per Backend (Plebs)
15.0
12.5
10.0
7.5
5.0
2.5
prod-09
prod-11
Concurrent Sessions per Backend (Panda)
30.0
25.0
20.0
15.0
10.0
5.0
prod-01
prod-02
prod-03
prod-04
prod-10
Request Rate per Frontend
Hz
600.0
480.0
360.0
240.0
120.0
rucio-lb (avg: 105.03914405)
rucio-lb-httponly (avg: 133.381524008)
Avg. Response Time per Backend (Plebs)
ms
160.0
140.0
120.0
100.0
80.0
60.0
40.0
20.0
Avg. Response Time per Backend (Panda)
250.0
200.0
150.0
100.0
50.0

#monitoringsucks

- Trending in 2011
- Cloud (r)evolution
  - Scaling up, automation
  - Servers as pets -> servers as cattle
- Many tools, not suited for new needs
  - Not scalable
  - Manual approach to system administration
  - Too many alerts

Characteristics

- Graphing/alerting are separate
- Manual approach to system administration
- No specific tools for log mgmt (tail, grep, awk!)

Monitoring in a DevOps world

- Integrated
  - metrics, logging, alerting, visualisation
- Self service
  - Service discovery
- Dynamic
  - New nodes, destroyed nodes, ...
- Collaborative
  - Shared between all disciplines within the team
- Observable applications
  - Applications provide their own instrumentation
  - Devs should implement this!

Types of monitoring

- Synthetic monitoring
- End user (experience) monitoring
- System monitoring
- Network monitoring
- Software metrics
- Application monitoring
- Log monitoring

Synthetic monitoring

- = generate traffic that simulates user interaction
- Is the application available for users?
- Simulate interactions programmatically
- Generates extra load!

Simple synthetic monitoring

```
$ time curl -I -s https://hogent.be/
HTTP/2 200
date: Fri, 28 Oct 2022 10:54:36 GMT
content-type: text/html;charset=UTF-8
vary: Accept-Encoding
...

real	0m0,235s
user	0m0,097s
sys	0m0,019s
```

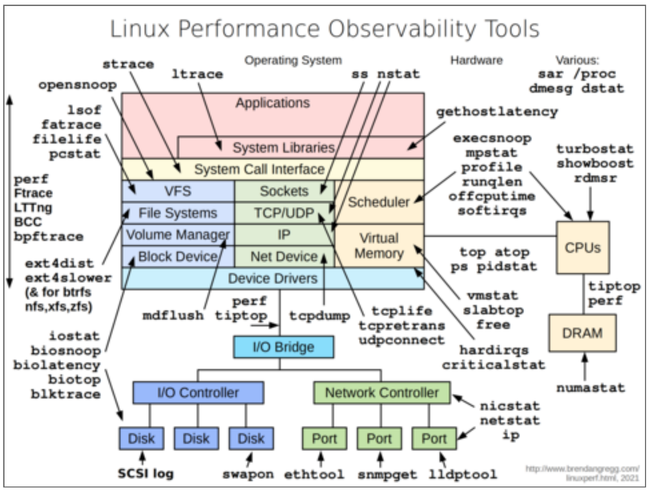
End user monitoring

- = gather data from real end user activity
    - Response times
    - Error rates
    - Number of user interactions
    - User metadata (device type, OS version, geolocation, ...)
- More variation in interactions
- No extra load

System monitoring

- This is what we are most familiar with!
- CPU, Memory, Disk I/O, Network I/O, ...
- Per process/per container/per VM metrics
- Kernel metrics, syscalls, ...
  System monitoring does not show whether your application is available to users!

System monitoring tools



Linux Performance Observability Tools Overview

Source: Gregg, B. (2021) *Linux Performance.*

https://www.brendangregg.com/linuxperf.html

Network monitoring

- Simple Network Management Protocol (SNMP)
  - Implemented in most network devices!
- Key metrics:
  - bandwith utilization
  - packet processing rates
  - latency
  - interface error rates
- Should be observed for every device on routing path of service

Software metrics

- Some 3rd party applications already expose performance metrics
- Not standardized, different for each application
- Usually, plugins exist to integrate in your monitoring solution

Application Monitoring

i.e. the application that your customers use and that your business generates income from!

- Often blind spot!
- Application problems:
  - App logic flaws
  - Capacity constraints
  - Dependency failures
- Measuring app health:
  - Transaction throughput (e.g. #completed reqs/s)
  - Response times
  - Error rates (e.g. HTTP 2xx, 4xx, 5xx)

Log monitoring

- System and application logs
- Often best way to determine what went wrong and to find how to fix it
- Application logs only work if devs have the app generate logs!
- Has its own set of specialised tooling (e.g. ELK stack, Splunk, Grafana Loki, ...)

> *I find that putting [devs] on call and making them figure out what’s going on in their application from their own logs, is the absolute fastest way to improve their logging.*
> *–Ernest Mueller (Director of Engineering, Six Nines IT)*

Putting it all together

- Enormous amount of metrics can be monitored
- Granularity also has impact
  - every 1s, 10s, 1m, …?
- Determine what is most important for your situation
- Don’t forget about application monitoring
  - Devs should make their apps observable!
- Share dashboards with application owners

Prometheus

Prometheus overview

https://prometheus.io/

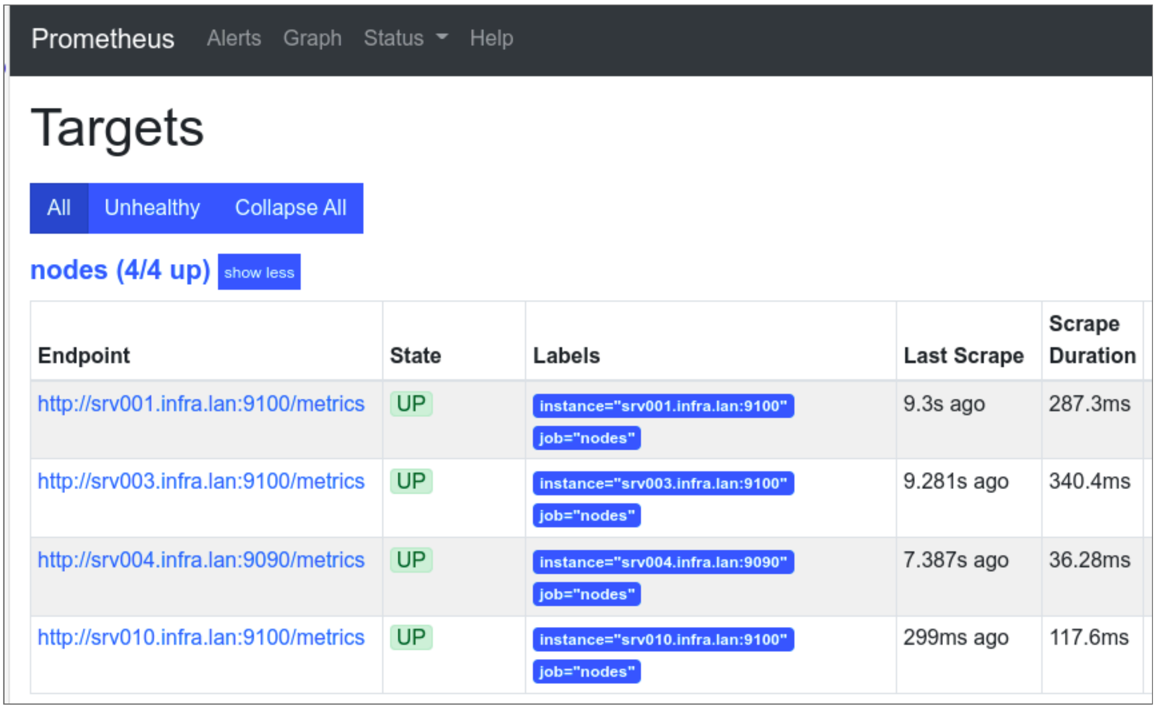
- Open source systems monitoring & alerting
- Originally from SoundCloud
- Now part of Cloud Native Computing Foundation
  - Home of Kubernetes
- Collects and stores metrics as time series data
  - time stamp
  - key-value pairs called labels

Features

- Time series collections via pull model over HTTP
- PromQL query language
- Service discovery or static configuration
- No reliance on distributed storage
- Multiple modes of graphing, dashboarding

Components

- Prometheus server:
  - scrapes, stores TS data
- **Client libraries**:
  - for instrumenting application code
- **Exporters**
  - for exposing metrics so Prometheus can scrape them
  - e.g. Node exporter for basic system metrics
  - also: MySQL, Kubernetes, SNMP, …
- AlertManager
- …



Targets

nodes (4/4 up)

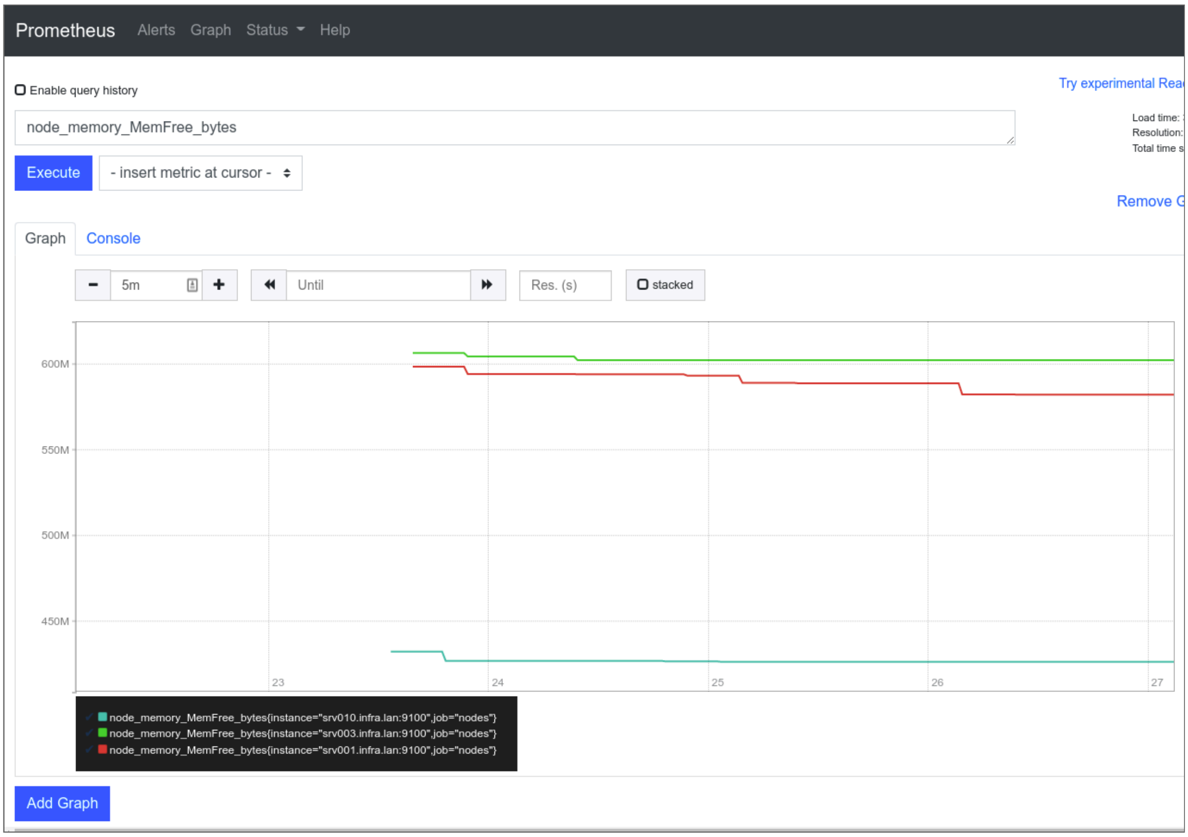
Endpoint	State	Labels	Last Scrape	Scrape Duration

http://srv001.infra.lan:9100/metrics	UP	instance="srv001.infra.lan:9100"
		job="nodes"
			9.3s ago	287.3ms

http://srv003.infra.lan:9100/metrics	UP	instance="srv003.infra.lan:9100"
		job="nodes"
			9.281s ago	340.4ms

http://srv004.infra.lan:9090/metrics	UP	instance="srv004.infra.lan:9090"
		job="nodes"
			7.387s ago	36.28ms

http://srv010.infra.lan:9100/metrics	UP	instance="srv010.infra.lan:9100"
		job="nodes"
			299ms ago	117.6ms



```
node_memory_MemFree_bytes
```

```
✓ 🟩 node_memory_MemFree_bytes{instance="srv010.infra.lan:9100",job="nodes"}
✓ 🟩 node_memory_MemFree_bytes{instance="srv003.infra.lan:9100",job="nodes"}
✓ 🟥 node_memory_MemFree_bytes{instance="srv001.infra.lan:9100",job="nodes"}
```

Lab Assignment

- Install Prometheus collection for Ansible
  - Galaxy page: **https://galaxy.ansible.com/ui/repo/published/prometheus/promethe**
  - Docs: **https://prometheus-community.github.io/ansible/branch/main/**
- Install Node Exporter on all VMs
  - role `prometheus.prometheus.node_exporter`
- Install Prometheus on a new VM
  - srv004, 172.16.128.4
  - role `prometheus.prometheus.prometheus`
- Collect metrics
- Set up Grafana dashboard