Traditional software monitoring

Why monitoring?

- Regardless of how experienced you are, when you build software, **problems will occur**.
- In order to deal with problems, you need to **be aware** of them and be able to **diagnose** what goes wrong.
- **Monitoring** your application is a way to do that.

Why monitoring?

- Your monitoring is a valuable **source of information** on which you can decide **how to act upon a problem** in the system.
- In traditional software monitoring there are different levels, typically divined as:
  - Level 1: Critical
  - Level 2: Major
  - Level 3: Minor

Why monitoring?

Level 1: Critical

Your authorization microservice is down, so no user is able to log in. This requires immediate action and is often monitored by a 24/7 alert/response team.

Why monitoring?

Level 2: Major

Your model is not able to make real time predictions anymore. It's a good thing that you implemented a fall back mechanism with a simple decision tree of business rules. However, it is important to fix the model as it performs a lot better than the fallback.

Why monitoring?

Level 3: Minor

You discover a bug in the system where 1% of the users have significantly larger loading time, because your systems gets into a strange loop. You can work on improving this bug on a scheduled basis.

Why monitoring?

You will be happy when you have monitoring in place when:

- incidents happen
- performance reviews (ability to back-track performance)

Why monitoring?

In software engineering there are 4 golden signals that should always be monitored:

- Traffic count
- Latency
- Errors
- Saturation

Why monitoring?

- Best practice to first build traditional software monitoring
- Then build ML specific monitoring on top
- *What metrics could be important to monitor in an ML system?*

Monitoring how?

We need something to:

- Remember the state of things we want to track (metrics)
- Alert when something is wrong
- Visualize the current and past state

Monitoring how?

Prometheus is a free software application used for event monitoring and alerting. It records real-time metrics in a time series database built using a HTTP pull model, with flexible queries and real-time alerting.

– Prometheus

Prometheus

- Open source, written in Go
- Time series based monitoring
- PromQL query language
- HTTP interface
- Multi-dimensional data model
- Python client

Prometheus

Data types / metrics

Prometheus has 4 default data types:

- Counter
- Gauge
- Histogram
- Summary

Prometheus

PromQL

Prometheus Query Language **PromQL**

Specific query language for prometheus data.

Prometheus

push vs pull

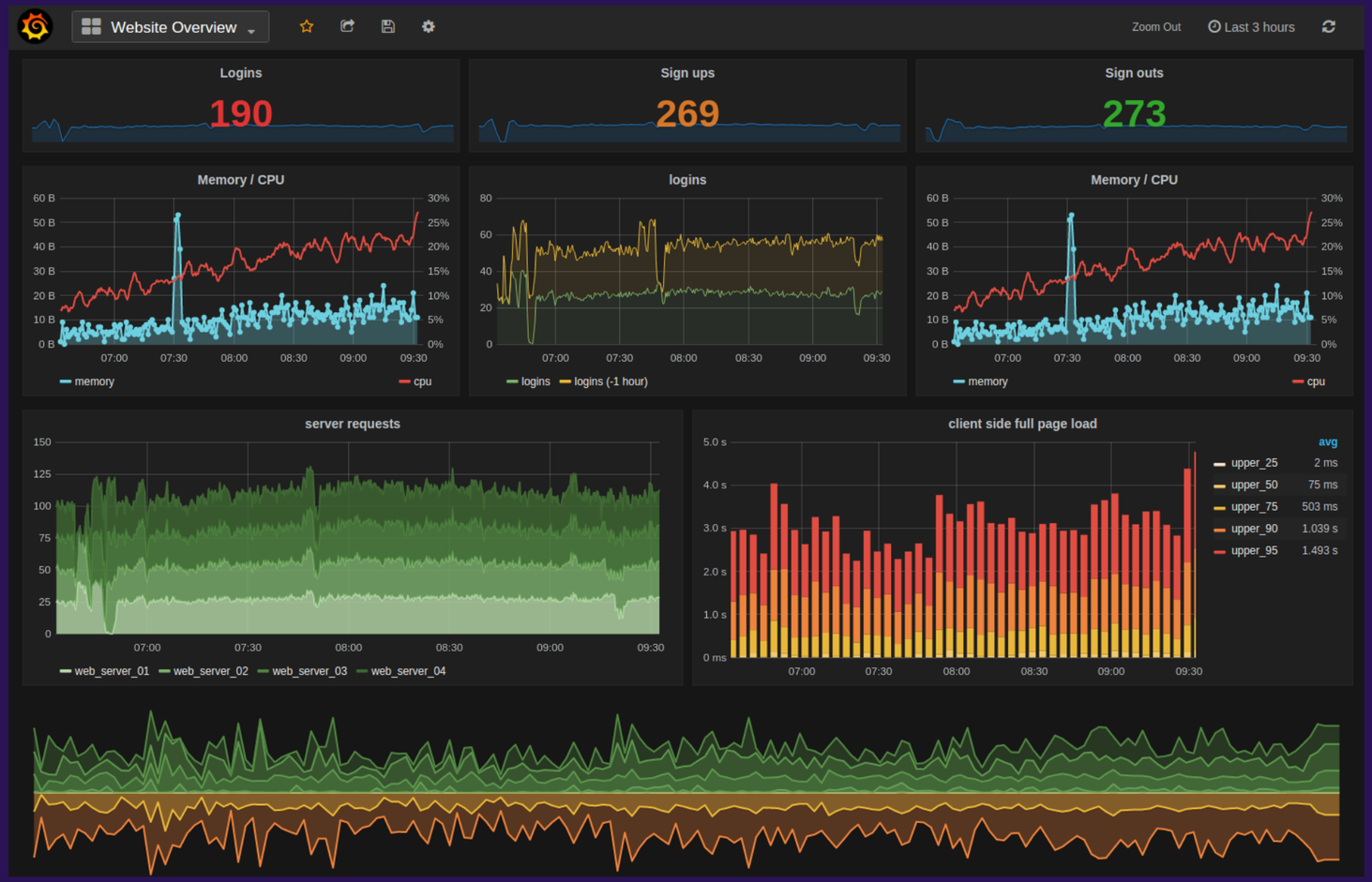
What are the advantages push- vs pull-based monitoring?

Monitoring how?

Grafana is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources.

– Grafana

Grafana


Website Overview
Zoom Out
Last 3 hours
Logins
190
Sign ups
269
Sign outs
273
Memory / CPU
logins
Memory / CPU
memory
cpu
logins
logins (-1 hour)
memory
cpu
server requests
web_server_01
web_server_02
web_server_03
web_server_04
client side full page load
avg
upper_25 2 ms
upper_50 75 ms
upper_75 503 ms
upper_90 1.039 s
upper_95 1.493 s

Project

Introduction

- We are going to build monitoring into a simulated system.
- I will provide starter code where there is a `/random` API endpoint that returns a random number with a certain delay.
- Today's goal is to build monitoring on the random endpoint.
- Later we will continue to build ML monitoring into this simulated system.

Project

Tech stack

- FastAPI
- Prometheus
- Grafana

To get up and running:

```
docker-compose build && docker-compose up
```

Project

Today's exercises

Provided the docker-compose file, the basic FastAPI app and an example of one metric:

1. **Create metrics** for the golden signals
2. **Track** the golden signals in **prometheus**
3. **Build** a dashboard in **Grafana** to show the golden signals

Project

End-of-day questions

- What is the difference between `rate` and `increase` in PromQL?
- What metrics would you monitor for an ML system?

Project

Next time

See if you can recover (with monitoring):

- the distribution of the random variable
- the distribution of the delay

Bonus:

- add metrics you find valuable for ML systems

References

What I learned from monitoring
more than 30 Machine Learning Use Cases
– PyData talk by Lina Weichbrodt