IDVOC
– observability

CRI
CENTRE DES RESSOURCES INFORMATIQUES

-- Cyril zarak Duval, root CRI/ACU 2020

DevOps and observability

DevOps is also about visibility for everyone

What is observability

- ➔ Once an application is in production, how does it behave ?
- ➔ Is it overloaded ?
- ➔ Is it working well ?
- ➔ Are there clients on it ?
- ➔ Are they facing errors ?
- ➔ Bugs ?
- ➔ If so, what kind ?
- ➔ How to investigate easily ?
- ➔ Shall we consider scaling up/down ?
- ➔ For an on-call ops, how to understand what's going on ?

What is observability

- A solution to all these questions are observability
- 3 pillars:
  - Monitoring
  - Logging
  - Tracing
- Ops shall provide platforms to receive these signals
- Dev shall provide such observability in their apps
  - And if applicable, documentation about the observability
    - And the actions to take if any

What is observability – logging

- Logs are the most useful indication to understand what is going on in the app in details, with description
  - They don't provide any overview though
- Useful to get information about:
  - Understanding what's going wrong
  - Which client/route/component is:
    - Used
    - Not working
    - Hammered
  - What is the app doing

What is observability – logs

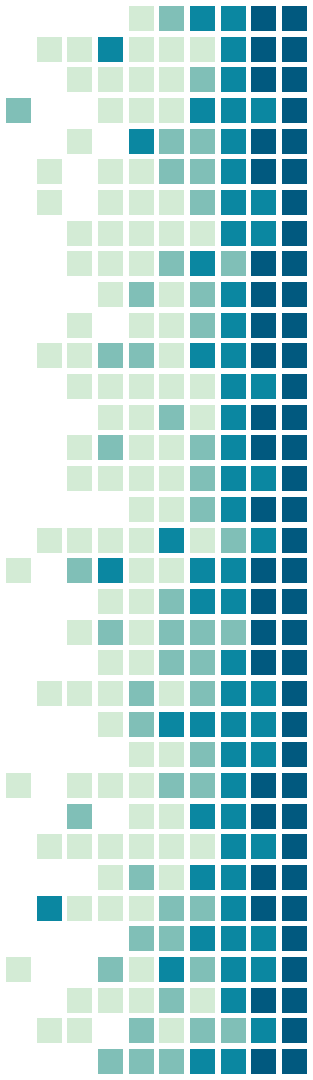
- Logs can hold a lot of value
  - Even legal one, mind the GDPR for example
- 2 schools of thoughts about providing logs:
  - stdout/stderr
  - syslog/elastic/… client
- **Logs must be structured**
  - syslog format
  - JSON
  - Homemade but consistent

What is observability - logs

- Why should logs be structured ?
- Useful to search for specific things
  - Logs will often be put in Loki, Elastic, …
  - They provide query languages
    - Ex: {component="auth", severity="error"}
    - Ex: client_id: 10 AND route: "/login"
- Having a structure (and a consistent and documented one) is important
- GiB of logs to be generated: not read manually

What is observability – logs

- Logs shall have a severity level:
  - DEBUG
  - INFO
  - WARNING
  - ERROR
- Severity level must be configurable
- The amount of logs generated must be chosen carefully
- For DEBUG, don't care
- Starting from INFO, one must be wise
- Use a logging library



What is observability – metrics

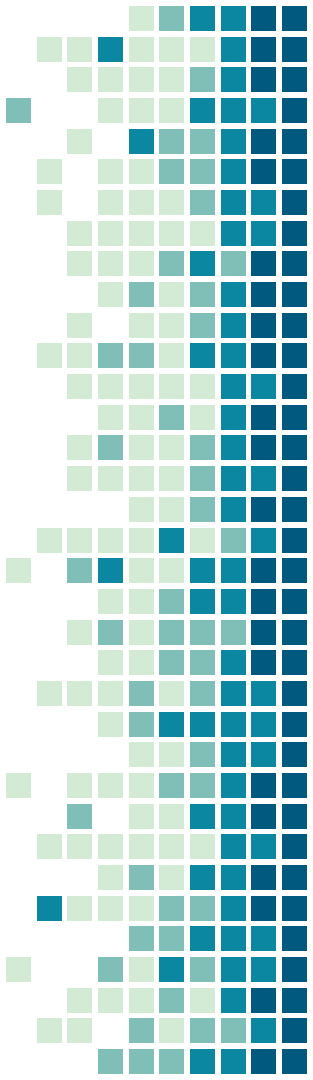
- Metrics is about exposing internal stats
- Metrics are to be collected by an external tool
  - Pull based, not push
- Metrics are usually meant to be plotted
- Most popular way of exposing metrics now:
  - Expose a HTTP route
    - /metrics
  - Prometheus format
- metric_name_unit{label1="value1", label2="value2"} value

What is observability – metrics

- What kind of metrics to expose ?
- total metrics
  - Number of requests handled in total
  - Number of file read in total
  - ….
- count/size/… metrics
  - Number of requests handled right now
  - Number of open files right now
  - Size of the event queue
  - …

What is observability - metrics

- What kind of metrics to expose ?
- seconds metrics
  - Amount of time taken to answer a request
  - Time taken writing data to cache
  - ....
- metadata metrics
  - Version of the running app
  - Running architecture
  - ...



What is observability - metrics

- Metrics are key to see what's going on
- We plot graph and we can visualize
- High level metrics (KPI) and low level
- Used for alerting
  - Ex: sudden drop of connected users
- Used for reporting
  - Ex: increasing amount of time taken to handle a request after an update
  - Ex: average user document size increasing over months

What is observability - alerting

- Observability is useful to see and try to understand what's going on with your app/infra
- Used for post-mortems
  - Evidence of the issue
  - Provide tools for RCA to try to determine the RCE
- If one can understand through metrics an issue that happened, why not alert when it happens ?
  - Or even before if we can

What is observability - alerting

- Observability is useful to see and try to understand what's going on with your app/infra
- Used for post-mortems
  - Evidence of the issue
  - Provide tools for RCA to try to determine the RCE
- If one can understand through metrics an issue that happened, why not alert when it happens ?
  - Or even before if we can

What is observability - alerting

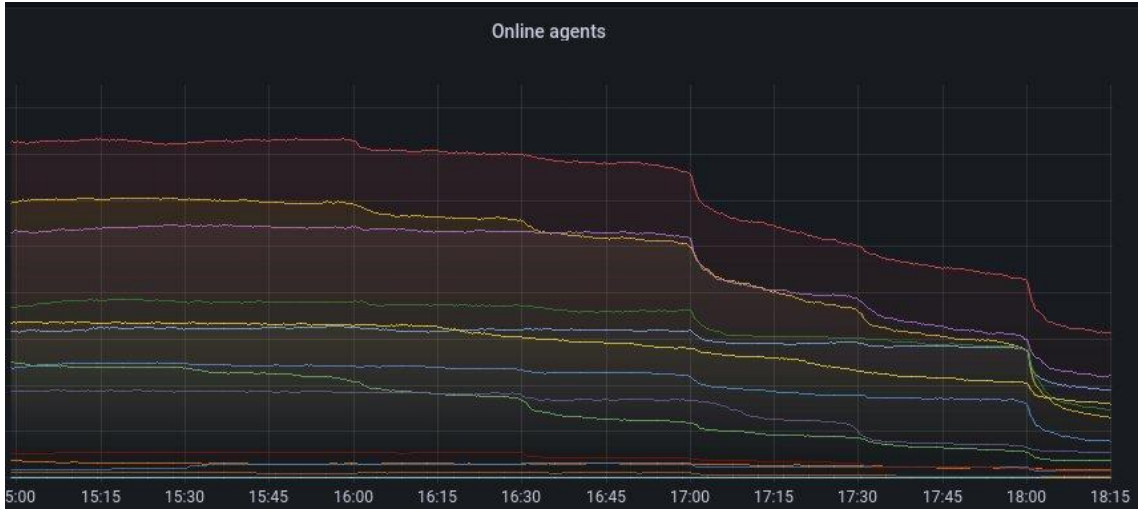
- ➔ Alerting should be done first on high-level metrics
  - ◆ Number of clients
  - ◆ Number of videos being watched
  - ◆ Number of emails sent
  - ◆ Latency increasing
- ➔ Alerting can be done on low-level metrics with care
  - ◆ If high CPU but no impact on the client, is it an alert ?
  - ◆ 10% of disk left, is it the same if 1GiB left of 1TiB ?

What is observability - alerting

- Alerts should have multiple level of criticality
  - Think about whether to wake up an ops or not for example
  - Lowest level(s) can even be moved to reporting
- Too many alerts = alert fatigue
- Too many false positives = more chances to miss an important one

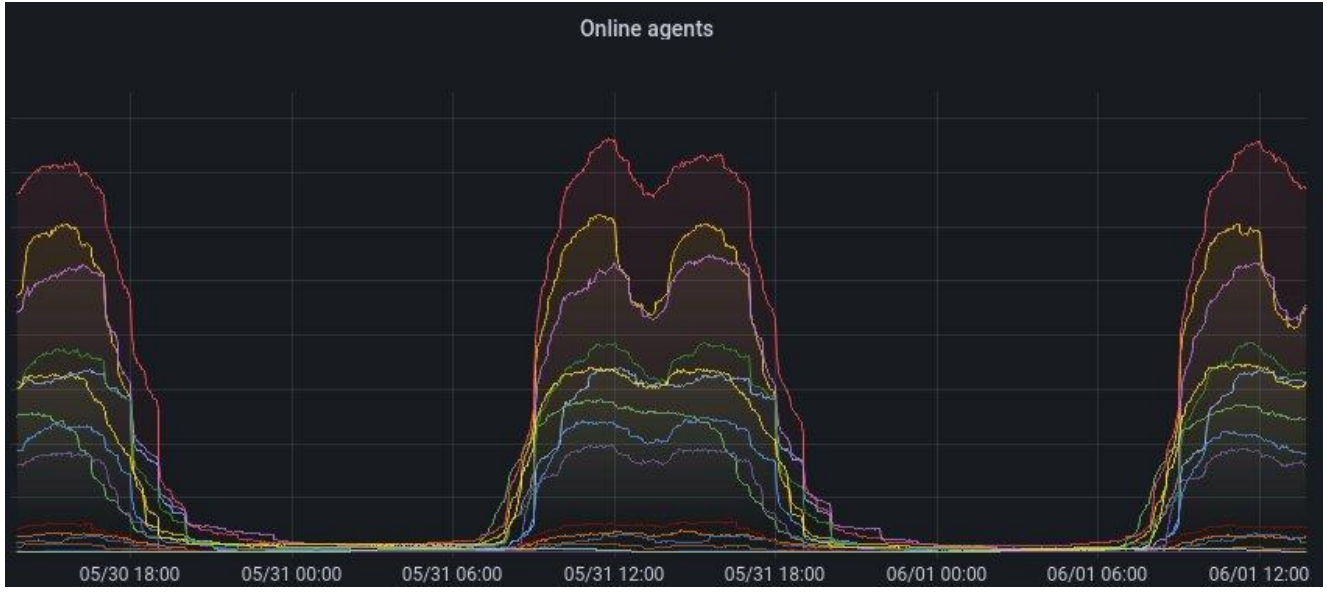
What is observability – alerting

➔ Be extra careful with alerting



What is observability – alerting

➔ Be extra careful with alerting



What is observability - error tracking

- What if you application throw an error ?
- You don't want the whole app to crash for most cases
- Just return an error to the client
- You also need the error to be reported to you to fix it
- Logs ?
  - Stacktrace are multi-lines
  - They have their own context
  - Put in the logs some concise information usually
    - ex: "Can't connect to DB"
    - ex: "Can't find <...> for <...> via <...>"

What is observability - error tracking

- Send the whole stacktrace and its context to another service
- Error tracking service
- For example sentry
- Regroup similar errors and plot their occurence
- Integrated with Gitlab to report bugs and regression
- Provide context
  - Browser used, account id, … if useful
  - Crumbs
  - Runtime data
  - Alerts

Thanks !

Questions ?

Slides available on zarak.fr/

Contact: cyril@cri.epita.fr

zarak production#5492