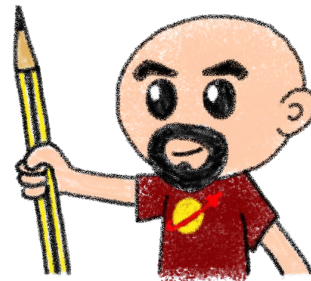




DEVOPS.BARCELONA

# Monitoring OVH: 300k servers, 28 DCs... and one Metrics platform

Horacio Gonzalez  
@LostInBrittany



# Who are we?

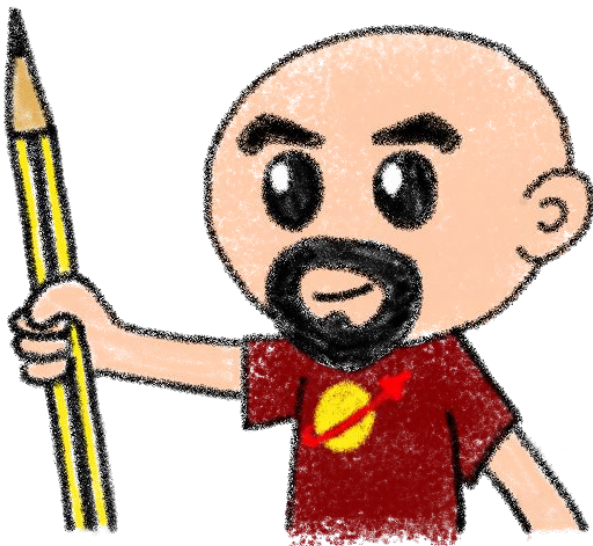
---

**Introducing myself and  
introducing OVH**

# Horacio Gonzalez

@LostInBrittany

Spaniard lost in Brittany, developer,  
dreamer and all-around geek







# OVH : Key Figures

**1.3M** Customers worldwide in **138** Countries  
**1.5 Billions euros** investment over five years  
**28** Datacenters (growing)  
**350k** Dedicated Servers  
**200k** Private cloud VMs running  
**650k** Public cloud Instances created in a month  
**20TB** bandwidth capacity  
**35** Points of presence  
**4TB** Anti DDoS capacity  
Hosting capacity : **1.3M** Physical Servers

+ **2 500** Employees in **19** countries  
**20** Years of Innovation

# OVH: Our solutions

 <b>Cloud</b>	 <b>Mobile Hosting</b>	 <b>Web Hosting</b>	 <b>Telecom</b>
VPS	Containers	Domain names	VoIP
Public Cloud	Compute	Email	SMS/Fax
Private Cloud	Database	CDN	Virtual desktop
Serveur dédié	Object Storage	Web hosting	Cloud HubIC
Cloud Desktop	Securities	MS Office	Over theBox
Hybrid Cloud	Messaging	MS solutions	

# Once upon a time...

## Because I love telling tales



# This talk is about a tale...



A true one nevertheless

# And as in most tales



It begins with a mission



# And a band of heroes



Engulfed into the adventure

# They fight against mishaps



And all kind of foes

# They build mighty fortresses



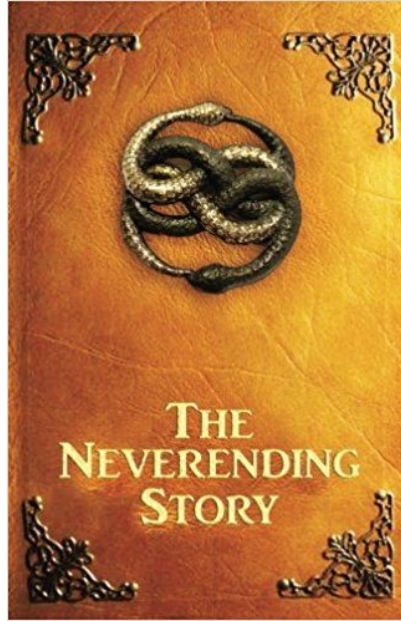
Pushing the limits of possible

# And defend them day after day



## Against all odds

# But we don't know yet the end



## Because this tale isn't finished yet

# It begins with a mission

---

Build a metrics platform for OVH

# A long time ago...

The screenshot shows the Thruk web interface. The main content area displays a table titled "Service Status Details For All Hosts". The table has columns for Host, Service, Status, Last Checked, and Duration. The table is filtered to show 30 items. A modal window is open over the table, showing a command "mschedule next check" and a "submit" button. Blue arrows point from the table to the modal and vice versa.

Host	Service	Status	Last Checked	Duration
n6_test_host_00	n6_test_ok_0	OK	11:30:46	172d 16h 44m 5s
n6_test_host_00	n6_test_ok_1	OK	11:30:46	172d 16h 34m 5s
n6_test_host_00	n6_test_ok_2	OK	11:30:46	172d 16h 24m 5s
n6_test_host_01	n6_test_ok_0	OK	11:30:46	172d 16h 43m 25s
n6_test_host_01	n6_test_ok_1	OK	2011-04-21 18:00:46	14d 17h 35m 55s
n6_test_host_01	n6_test_ok_2	OK	2011-04-21 18:00:46	14d 17h 35m 55s
n6_test_host_02	n6_test_ok_0	OK	11:30:46	172d 16h 42m 45s
n6_test_host_02	n6_test_ok_1	OK	11:30:46	172d 16h 32m 45s
n6_test_host_02	n6_test_ok_2	OK	11:30:46	172d 16h 22m 45s
n6_test_host_03	n6_test_ok_0	OK	11:30:46	172d 16h 42m 5s
n6_test_host_03	n6_test_ok_1	OK	11:07:55	172d 16h 32m 5s
n6_test_host_03	n6_test_ok_2	OK	11:29:46	172d 16h 22m 5s
n6_test_host_04	n6_test_ok_0	OK	11:28:13	172d 16h 41m 25s
n6_test_host_04	n6_test_ok_1	OK	11:06:13	172d 16h 31m 25s
n6_test_host_04	n6_test_ok_2	OK	11:10:01	172d 16h 21m 25s
n6_test_host_05	n6_test_ok_0	OK	11:28:32	172d 16h 40m 45s

# A long time ago...

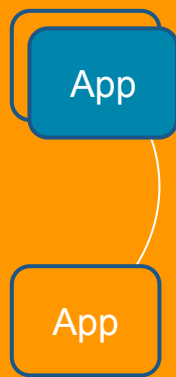
Monitoring: **Does** the system works?



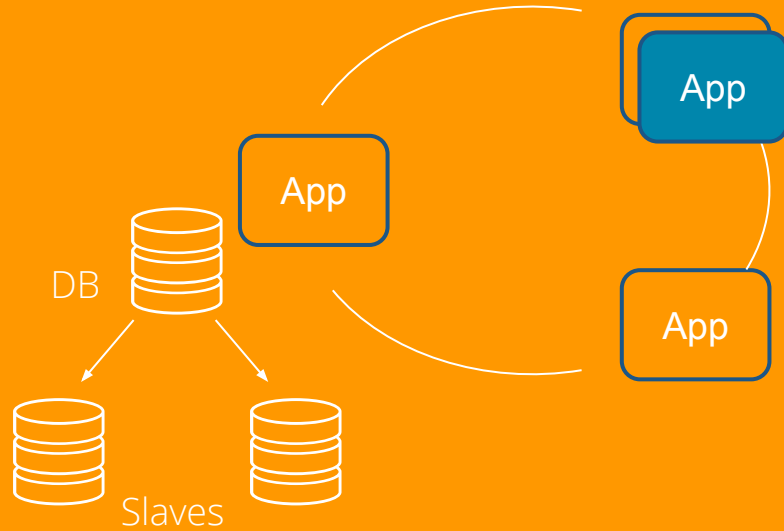
# Moving from monolith to $\mu$ services

App

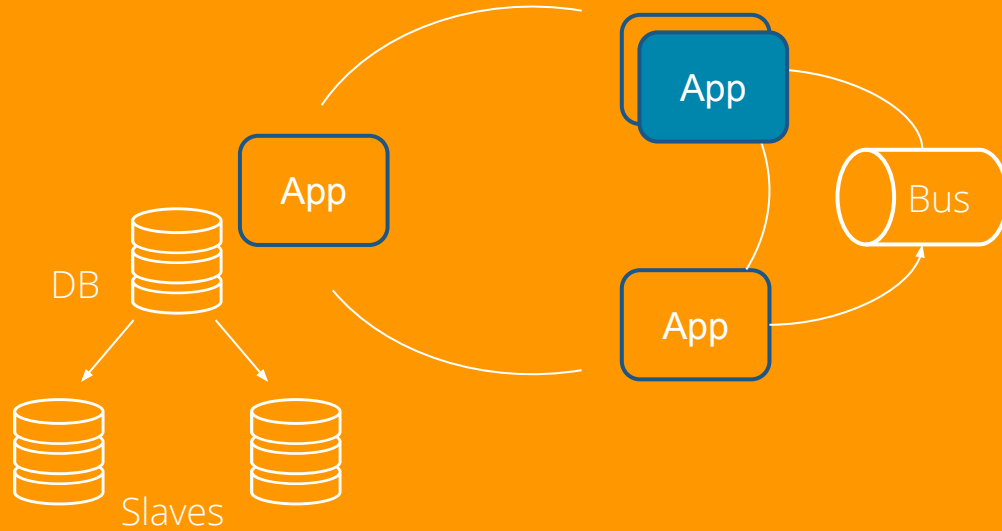
# Moving from monolith to $\mu$ services



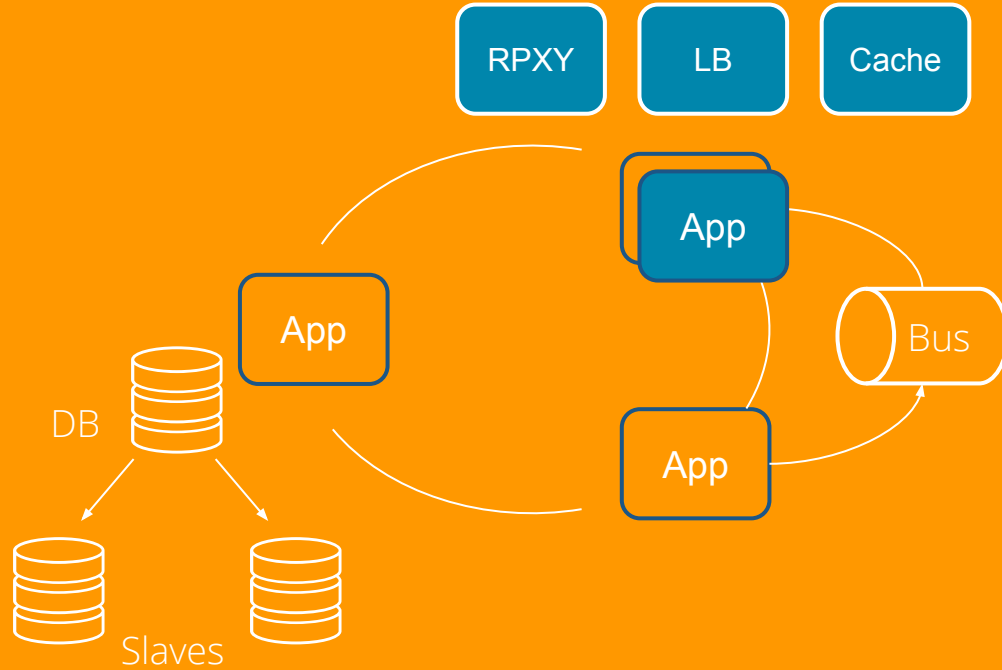
# Moving from monolith to $\mu$ services



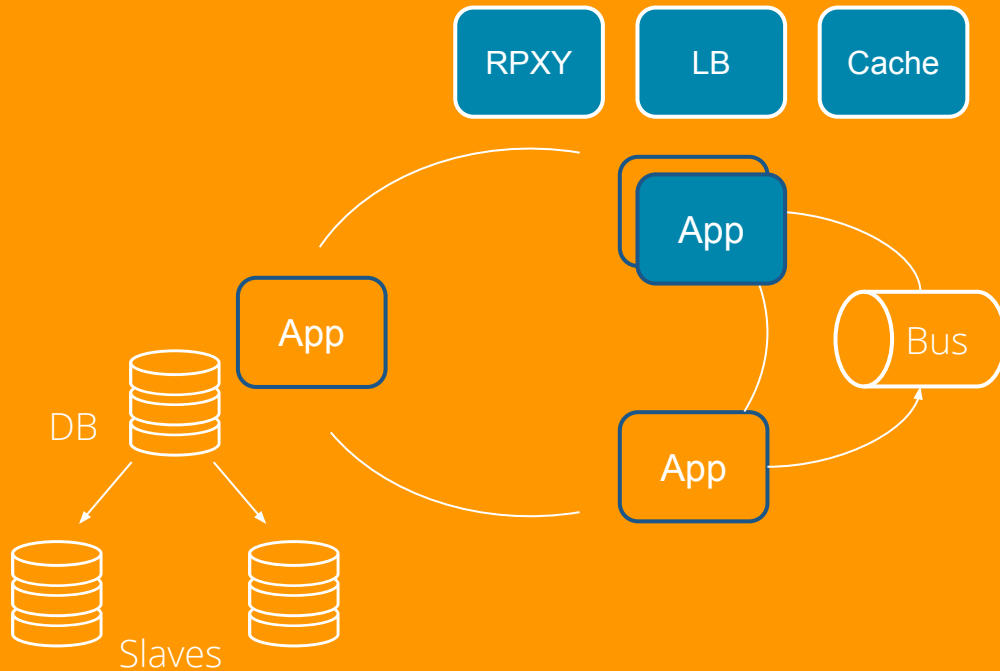
# Moving from monolith to $\mu$ services



# Moving from monolith to $\mu$ services



# What could go wrong?



# Microservices are a distributed system

The Microservices Complexity Paradox

+ Joyent



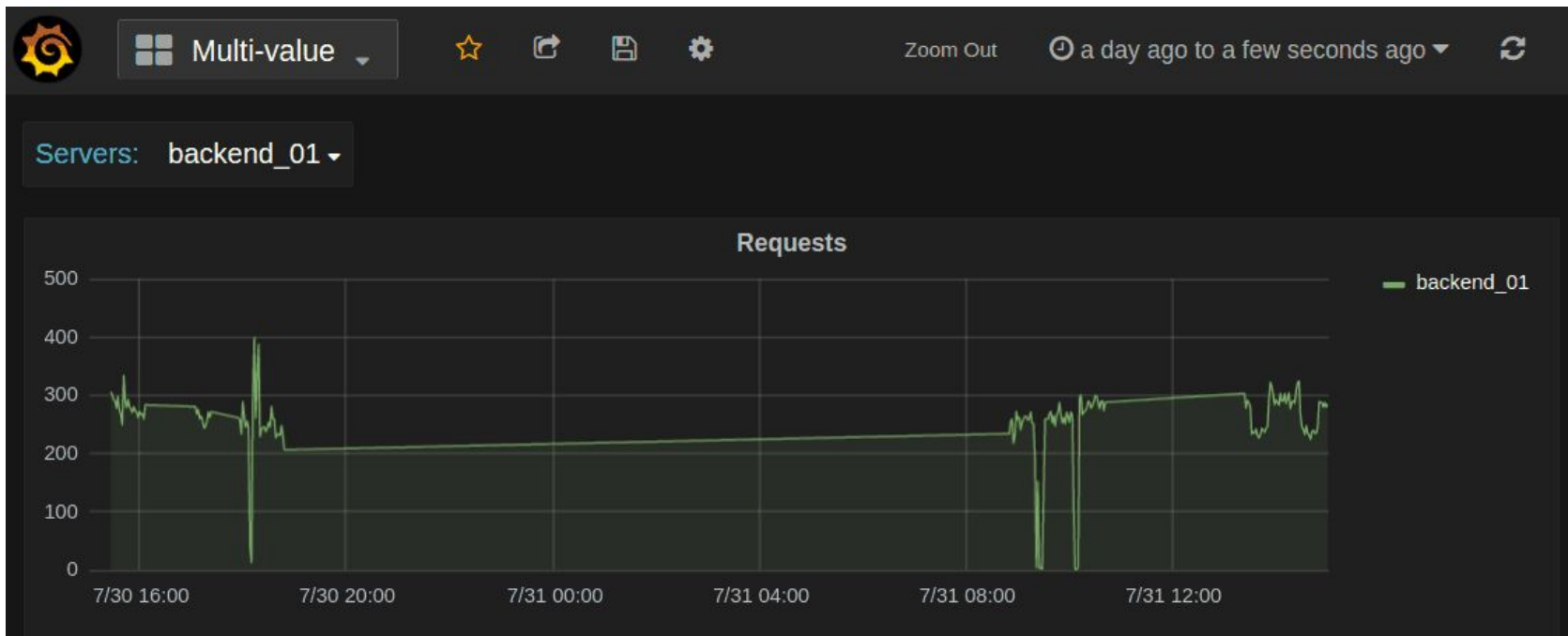
GOTO 2017 • Debugging Under Fire: Keep your Head when Systems have Lost their Mind • Bryan Cantrill

# We need to have insights

Observability : Understand **how** it works



# OVH decided go metrics-oriented



# A metrics platform for OVH



**For all OVH**

# Building OVH Metrics

---

One Platform to unify them all,  
One Platform to find them,  
One Platform to bring them all  
and in the Metrics monitor them



# What is OVH Metrics?

Managed Cloud Platform  
for Time Series

# OVH monitoring story

We had lots of partial solutions...



OPENTSDDB



mongoDB®



graphite



*influxdb*



# OVH monitoring story

One Platform to unify them all

What should we build it on?

# OVH monitoring story

Including a really big



# OpenTSDB drawbacks

## OpenTSDB RowKey Design

metrics timestamp tagk1 tagv1 tagk2 tagv2





# OpenTSDB Rowkey design flaws

- `.*regex.*` => full table scans
- High cardinality issues (Query latencies)



We needed something able to manage  
**hundreds of millions** time series

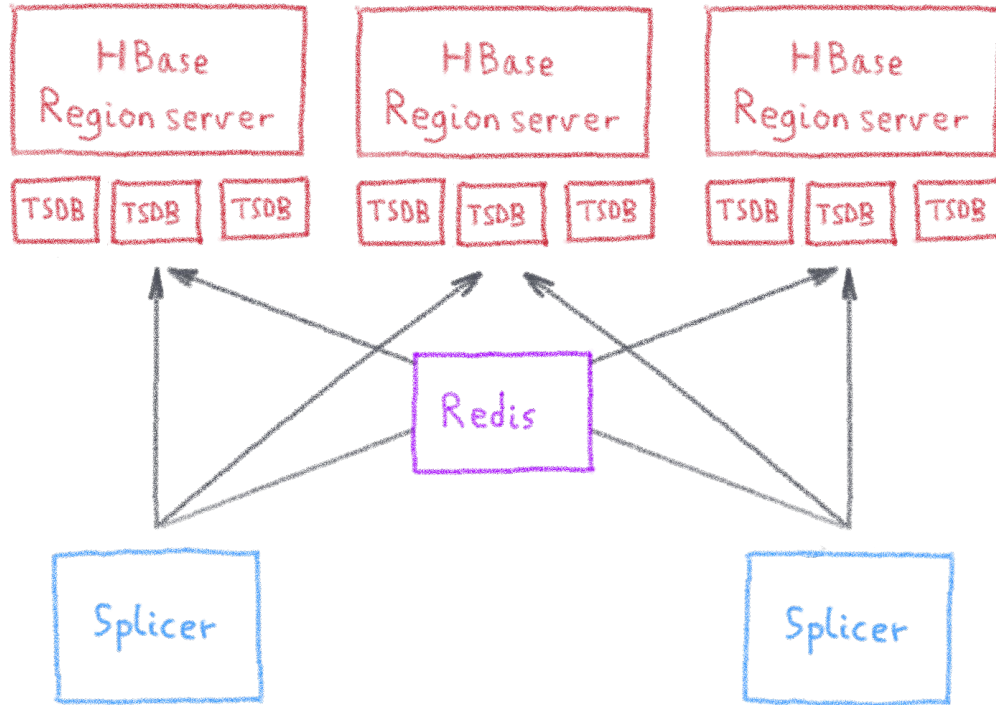


OpenTSDB didn't  for us

# OpenTSDB other flaws

- Compaction (or append writes)
- /api/query : 1 endpoint per function?
- Asynchronous
- Unauthenticated
- ...

# Scaling OpenTSDB



# Metrics needs

First **need**:

To be **massively** scalable

# Analytics is the key to success



Fetching data is only the tip of the iceberg

# Analysing metrics data



To be scalable, analysis must be done in the database, not in user's computer

# Metrics needs

Second **need**:

To have **rich query** capabilities

# Enter Warp 10...

Open-source  
Time series  
Database





# More than a Time Series DB

Warp 10 is a software platform that

- Ingests and stores time series
- Manipulates and analyzes time series



# Manipulating Time Series with Warp 10

## A true Time Series analysis toolbox

- Hundreds of functions
- Manipulation frameworks
- Analysis workflow

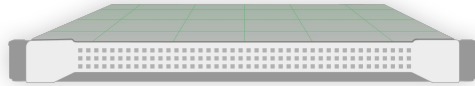
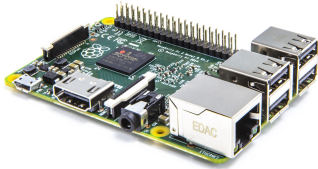


# Manipulating Time Series with Warp 10

A Time Series manipulation language



# Did you say scalability?

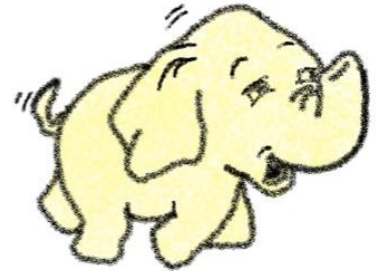


From the smallest to the largest...

# More Warp 10 goodness

- Secured & multi tenant
- In memory Index
- No cardinality issues
- Lockfree ingestion
- WarpScript Query Language
- Support more data types
- Synchronous (transactions)
- Better Performance
- Better Scalability
- Versatile  
(standalone, distributed)

# OVH Observability Metrics Platform



# Building an ecosystem

---

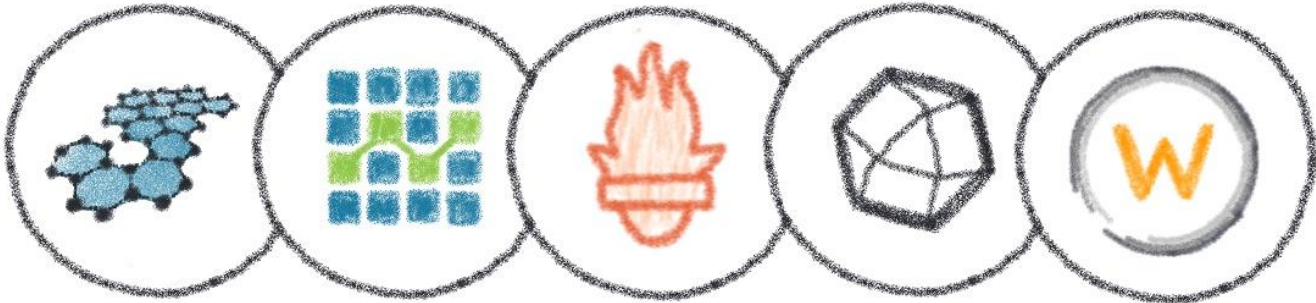
From Warp 10 to OVH Metrics

# What protocols should we support?

Who must do the effort?



# Open source monitoring tools



# Open source monitoring tools



# Open source monitoring tools



OPENTSDDB

# Open source monitoring tools



Prometheus

# Open source monitoring tools



# Open source monitoring tools



# Open source monitoring tools

Why choose?  
Let's support all of them!

# Metrics Platform

## Operators



Integrate with Operators to avoid pull/push of data

## Query



Query your data using any language among WarpScript, OpenTSDB, Prometheus and Graphite  
Visualize with Grafana



## Input

Ingest data using best fitted protocol among Warp10, OpenTSDB, Prometheus, InfluxData and Graphite - Datapoints are available with any Query protocol



## Automator

Register Loop queries to power your smart automation platform



# Metrics Platform

**graphite**

**influx**

https://

**opentsdb**

.<region>.metrics.ovh.net

**prometheus**

**Warp10**

**tsl**

...



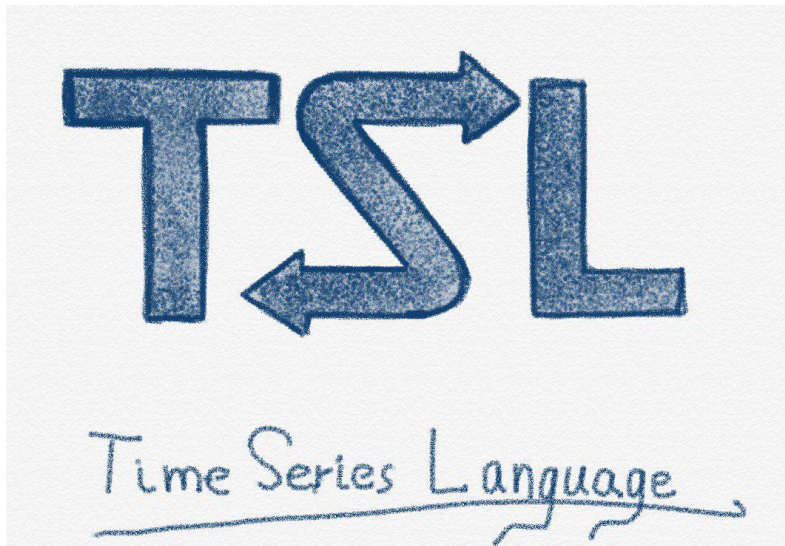
# Metrics Platform

https://  
graphite  
influx  
opentsdb  
prometheus  
Warp10  
tsl  
...

.<region>.metrics.ovh.net

# TSL

```
select("cpu.usage_system")  
  .where("cpu~cpu[0-7]*")  
  .last(12h)  
  .sampleBy(5m,max)  
  .groupBy(mean)  
  .rate()
```

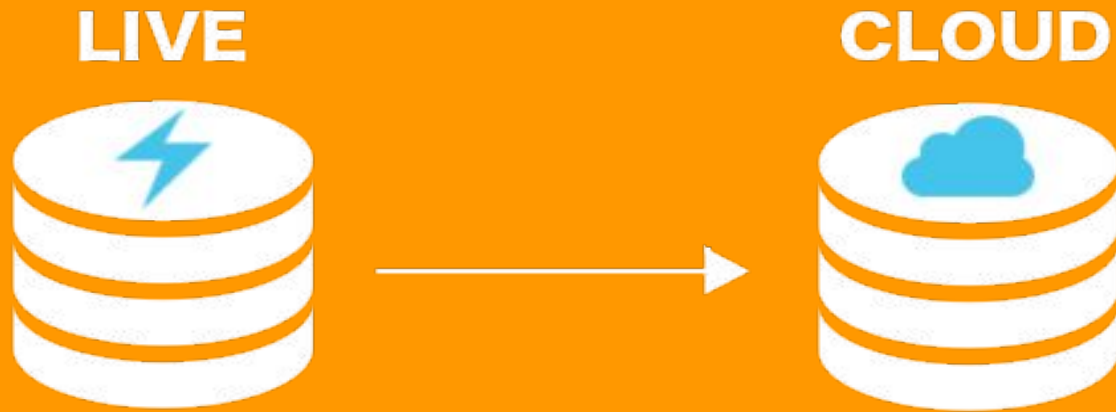


[github.com/ovh/tsl](https://github.com/ovh/tsl)

# Metrics Live

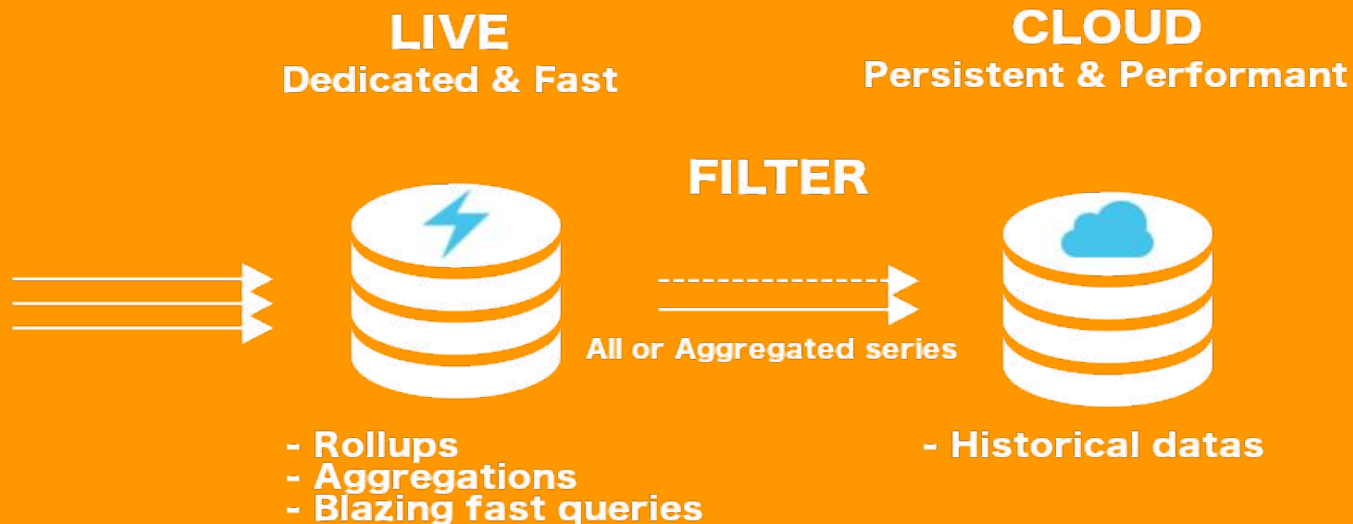
In-memory, high-performance Metrics instances

# In-memory: Metrics live



millions of writes/s

# In-memory: Metrics live



# In-memory: Metrics live

## STAGE 1

Short retention - hours  
Fine grained monitoring  
Raw data



## STAGE 2

Short retention - days  
Consolidated aggregations  
Global Infra monitoring



## STAGE 3

Customer metrics  
Historical datas

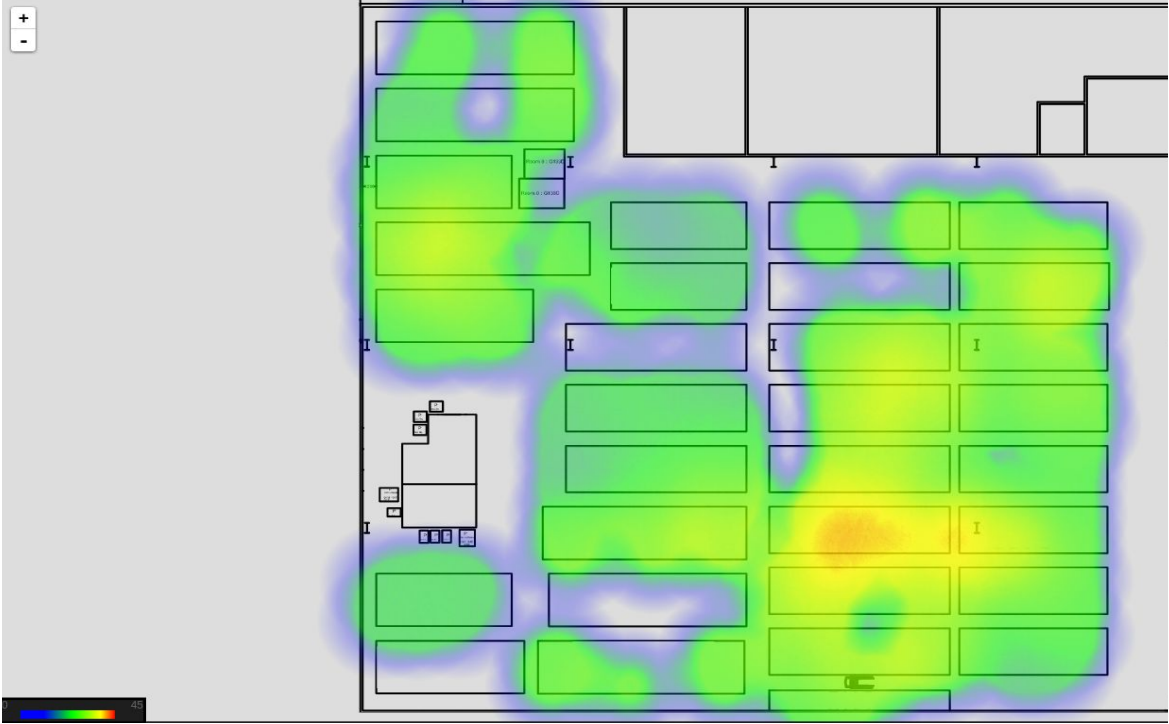


# Monitoring is only the beginning

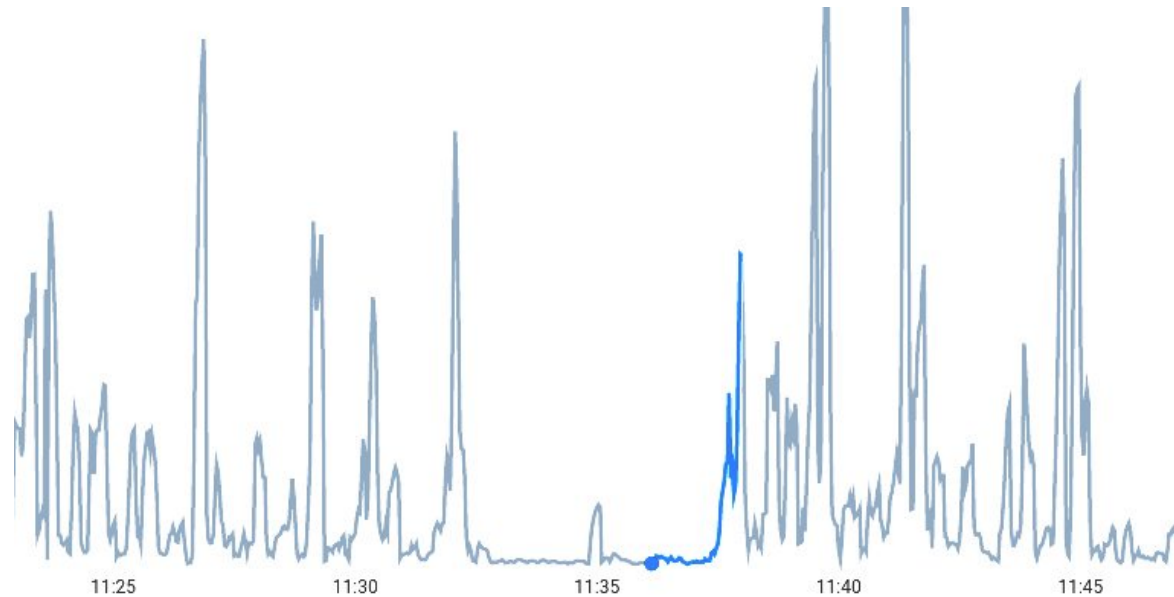
OVH Metrics answer to many other use cases



# Graveline rack's temperature



# Even medical research...



Metrics' Pattern Detection feature helped Gynaecology Research  
to prove patterns on perinatal mortality

# Use cases families

- Billing .....(e.g. bill on monthly max consumption)
- Monitoring .....(APM, infrastructure, appliances,...)
- IoT .....(Manage devices, operator integration, ...)
- Geo Location .....(Manage localized fleets)

# Use cases

---

- DC Temperature/Elec/Cooling map
- Pay as you go billing (PCI/IPLB)
- GSCAN
- Monitoring
- ML Model scoring (Anti-Fraude)
- Pattern Detection for medical applications

# SREing Metrics

---

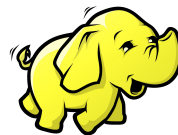
**With a great power  
comes a great responsibility**

# Metrics's metrics

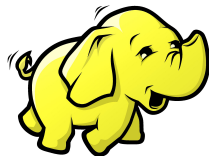
432.000.000.000  
datapoints / day

# Our stack overview

- More than **650** machines operated by 5 people
- **>95%** dedicated servers
- No Docker, only SystemD
- Running many Apache projects:
  - Hadoop
  - HBase
  - Zookeeper
  - Flink
- And Warp 10



# Our biggest Hadoop cluster



200 datanodes

2.3 PB of **capacity**  
8.5Gb/s of **bandwidth**



~60k regions of 10Gb

1.5M of **writes/s**  
3M of **reads/s**



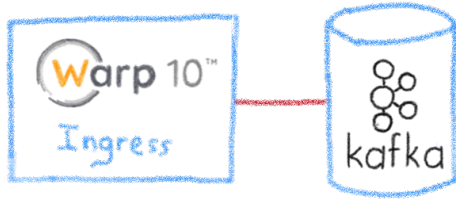
# Hadoop need a lot of ❤️



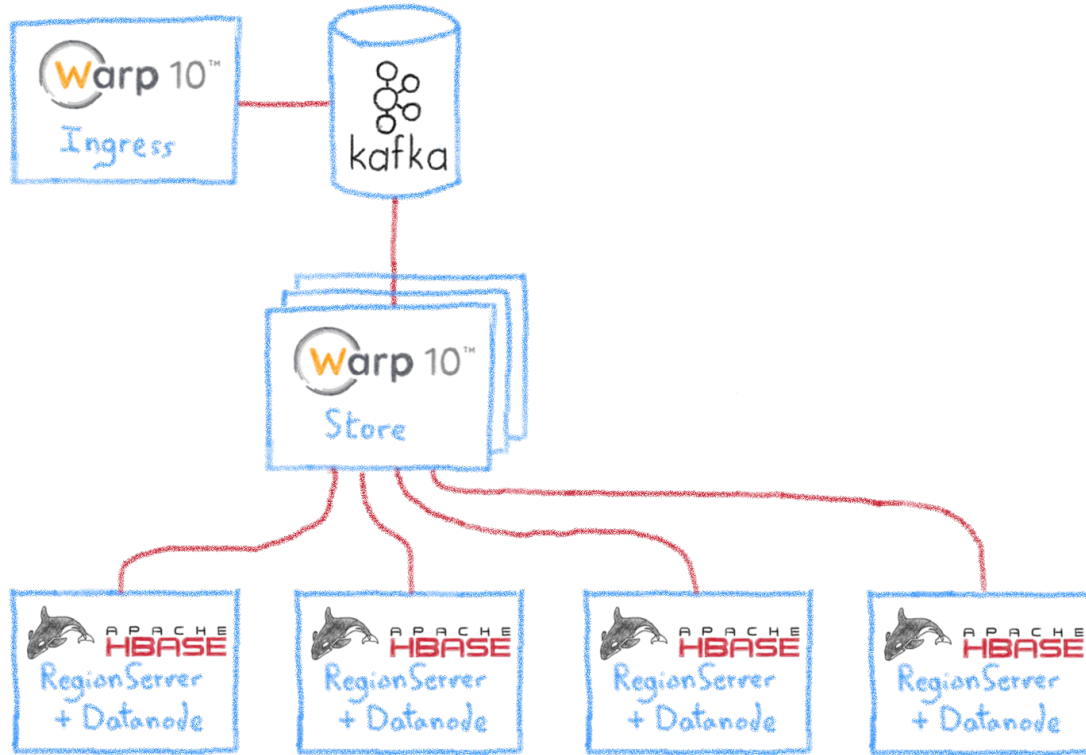
# Warp10: distributed overview



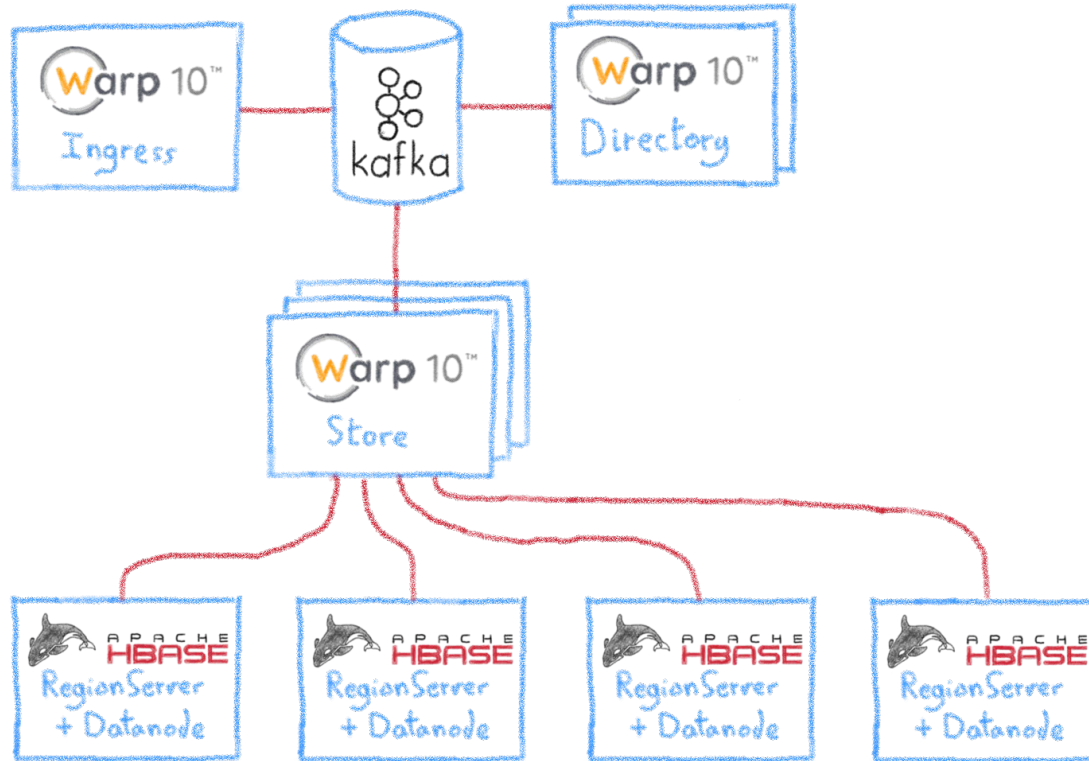
# Warp10: distributed overview



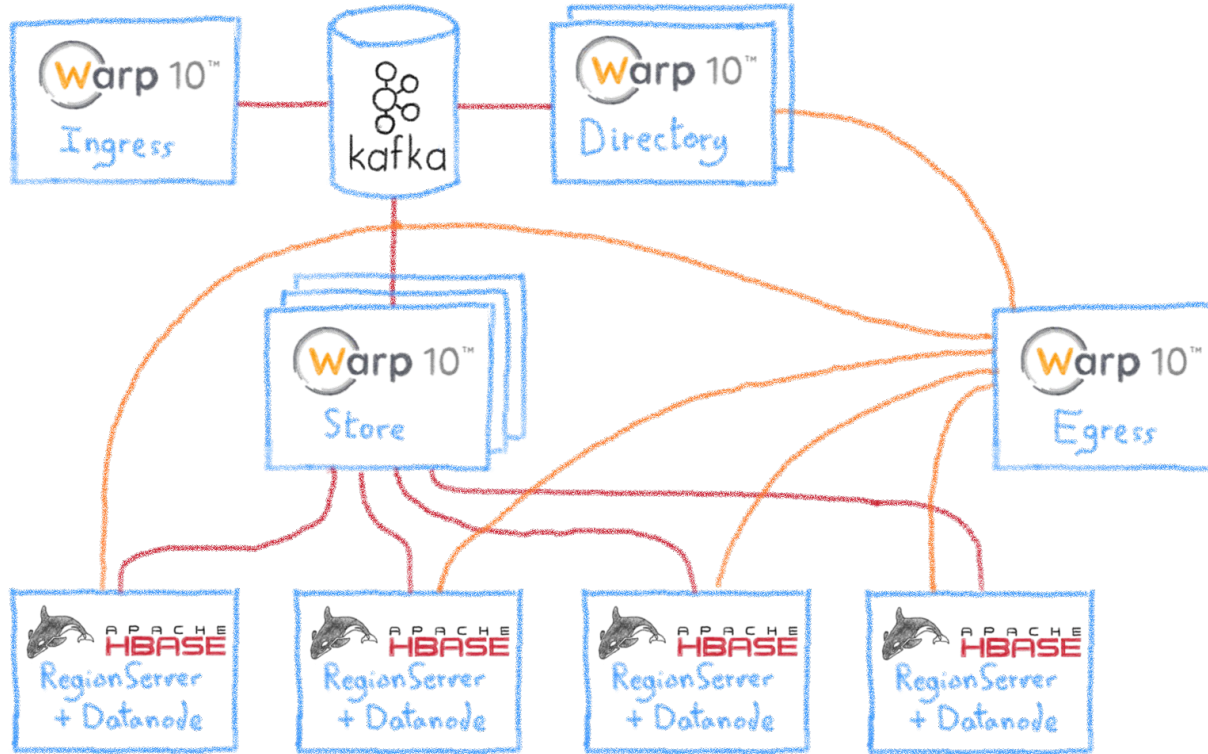
# Warp10: distributed overview



# Warp10: distributed overview



# Warp10: distributed overview



# Hadoop nodes

Most of the nodes are the following:

- 16 to 32 cores
- 64 to 128 GB of RAM 🤯
- 12 to 16 TB

But, we also have some huge nodes:

- 2x 20 cores (xeon gold)
- 320 GB of RAM 🤯 🤯
- 12x 4TB of Disk

# Warp10 nodes

## Ingress (cpu-bound):

- 32 cores
- 128 GB of RAM 🤯

## Directory (ram-bound):

- 48 cores
- 512 GB of RAM 🤯 🤯 🤯

## Egress (cpu-bound):

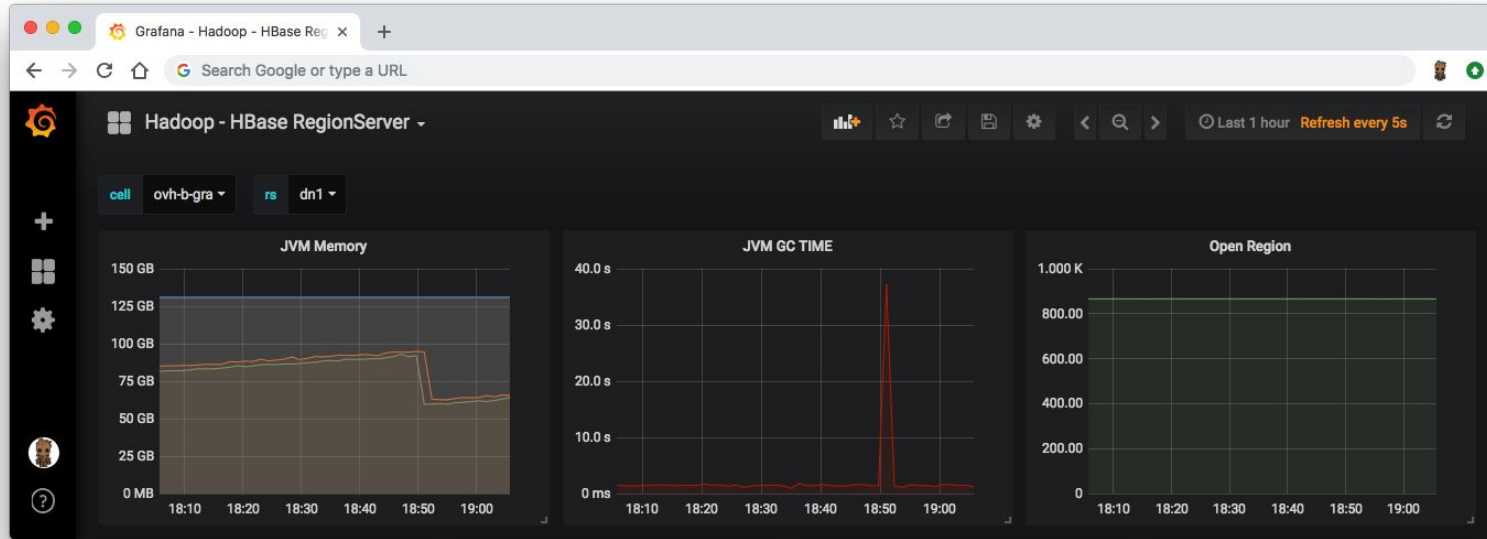
- 32 cores 🤯
- 128 GB of RAM

## Store (cpu-bound):

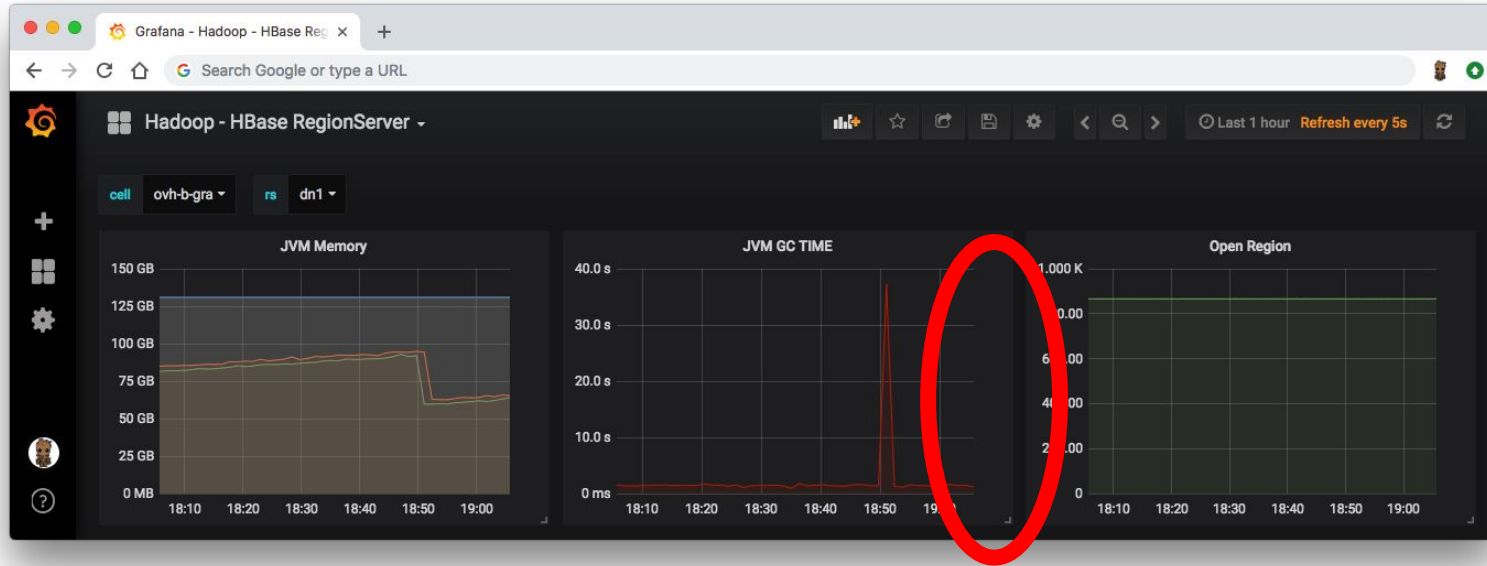
- 32 cores 🤯
- 128 GB of RAM



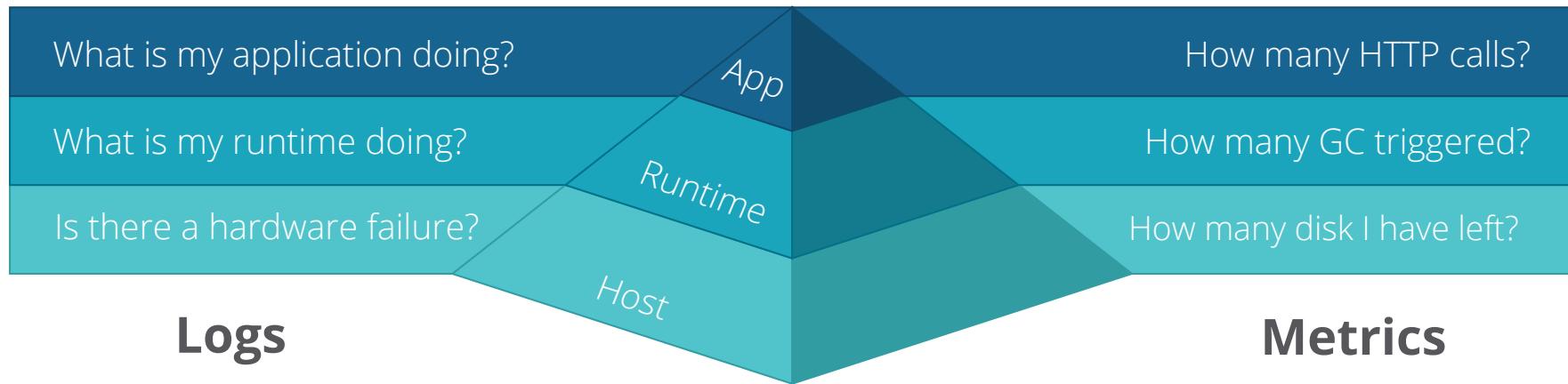
# Why you should care?



# Why you should care? (>30s) 🤯



# The only way to optimize: measure



# Monitoring JVM with metrics

prometheus / jmx\_exporter

Watch ▾

65

★ Star

852

Fork

398

<> Code

🔔 Issues 19

🔗 Pull requests 17

📁 Projects 0

📊 Insights

A process for exposing JMX Beans via HTTP for Prometheus consumption

jmx

prometheus

mbean

java-agent

monitoring

prometheus-exporter

📄 226 commits

🌿 1 branch

📦 13 releases

👤 60 contributors

📄 Apache-2.0

# Monitoring JVM with metrics

## Running

---

To run as a javaagent [download the jar](#) and run:

```
java -javaagent:./jmx_prometheus_javaagent-0.11.0.jar=8080:config.yaml -jar yourJar.jar
```

Metrics will now be accessible at <http://localhost:8080/metrics>

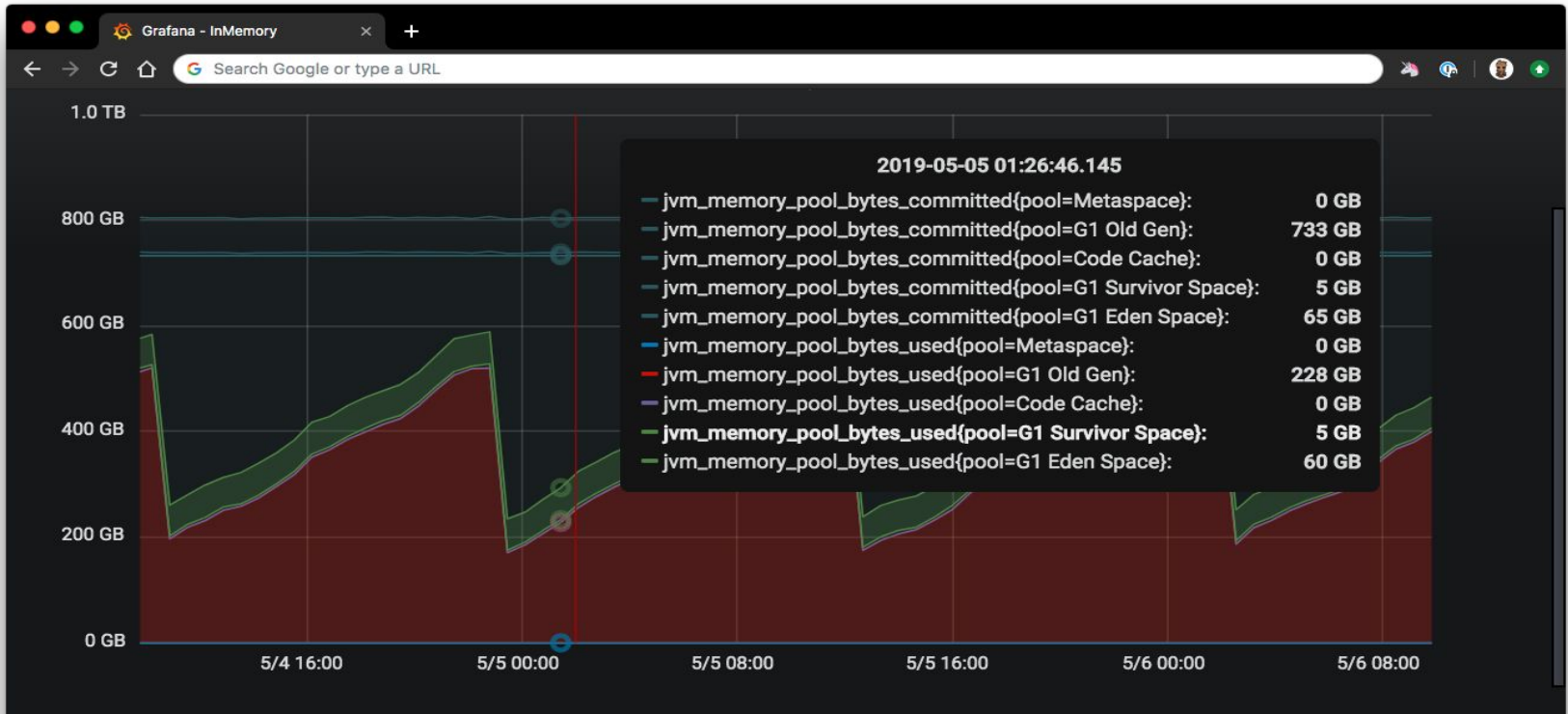
# Monitoring JVM with metrics

```
1. metrics@GW_IM: ~/ansible/ansible-warp10-standalone (ssh)
root@A.GRA:~# curl -s http://127.0.0.1:9101/metrics | grep -v "#"
process_cpu_seconds_total 1.029816855E8
process_start_time_seconds 1.522059928366E9
process_open_fds 109.0
process_max_fds 512000.0
process_virtual_memory_bytes 2.42578112512E11
process_resident_memory_bytes 2.41437425664E11
java_lang_memorypool_collectionusagethresholdsupported{name="Metaspace",} 0.0
java_lang_memorypool_collectionusagethresholdsupported{name="Code Cache",} 0.0
java_lang_memorypool_collectionusagethresholdsupported{name="G1 Eden Space",} 1.0
java_lang_memorypool_collectionusagethresholdsupported{name="G1 Old Gen",} 1.0
java_lang_memorypool_collectionusagethresholdsupported{name="G1 Survivor Space",} 1.0
java_lang_runtime_uptime 3.4834238296E10
java_lang_garbagecollector_lastgcinfo_memoryusagebeforegc_used{name="G1 Young Generation",key="G1 Survivor Space",} 1.711276032E9
java_lang_garbagecollector_lastgcinfo_memoryusagebeforegc_used{name="G1 Young Generation",key="Metaspace",} 3.1310464E7
java_lang_garbagecollector_lastgcinfo_memoryusagebeforegc_used{name="G1 Young Generation",key="G1 Old Gen",} 1.28463160496E11
java_lang_garbagecollector_lastgcinfo_memoryusagebeforegc_used{name="G1 Young Generation",key="G1 Eden Space",} 2.4058527744E10
java_lang_garbagecollector_lastgcinfo_memoryusagebeforegc_used{name="G1 Young Generation",key="Code Cache",} 3.813536E7
java_lang_memory_nonheapmemoryusage_init 4194304.0
java_lang_operatingsystem_committedvirtualmemorysize 2.42578120704E11
java_lang_memory_objectpendingfinalizationcount 0.0
java_lang_memorypool_collectionusagethresholdexceeded{name="G1 Eden Space",} 0.0
java_lang_memorypool_collectionusagethresholdexceeded{name="G1 Old Gen",} 0.0
```

# Monitoring JVM with metrics



# Monitoring JVM with metrics





# Tuning G1 is hard 🥲

```
-Xms800g -Xmx800g \  
-XX:+UseG1GC -XX:G1HeapRegionSize=64m \  
-XX:MaxGCPauseMillis=500 \  
-XX:ParallelGCThreads=36 \  
-XX:ConcGCThreads=9 \  
-XX:+UnlockExperimentalVMOptions \  
-XX:G1NewSizePercent=8 \  
-XX:G1MaxNewSizePercent=8 \  
-XX:+ParallelRefProcEnabled \  
-XX:+PerfDisableSharedMem \  
-XX:-ResizePLAB \  
-XX:-ReduceInitialCardMarks \  
-XX:G1RSetRegionEntries=4096 \  
-XX:InitiatingHeapOccupancyPercent=65 \  
-XX:G1HeapWastePercent=10 \  
-XX:G1MixedGCCountTarget=16 \  

```

# Tuning G1 is hard 🥲🥲

```
-Xms800g -Xmx800g \  
-XX:+UseG1GC -XX:G1HeapRegionSize=64m \  
-XX:MaxGCPauseMillis=500 \  
-XX:ParallelGCThreads=36 \  
-XX:ConcGCThreads=9 \  
-XX:+UnlockExperimentalVMOptions \  
-XX:G1NewSizePercent=8 \  
-XX:G1MaxNewSizePercent=8 \  
-XX:+ParallelRefProcEnabled \  
-XX:+PerfDisableSharedMem \  
-XX:-ResizePLAB \  
-XX:-ReduceInitialCardMarks \  
-XX:G1RSetRegionEntries=4096 \  
-XX:InitiatingHeapOccupancyPercent=65 \  
-XX:G1HeapWastePercent=10 \  
-XX:G1MixedGCCountTarget=16 \  

```

```
-XX:+HeapDumpOnOutOfMemoryError \  
-XX:HeapDumpPath=/opt/warp/logs/heap.dump \  
-verbose:gc \  
-XX:+PrintGC \  
-XX:+PrintGCDetails \  
-XX:+PrintGCDateStamps \  
-XX:+PrintGCTimeStamps \  
-Xloggc:/opt/warp/logs/gc.log \  
-XX:+UseGCLogFileRotation \  
-XX:NumberOfGCLogFiles=10 \  
-XX:GCLogFileSize=10M \  
  
-XX:+AlwaysPreTouch \  
-XX:+UseTransparentHugePages \  
-XX:+UseNUMA \  
-XX:-UseBiasedLocking \  

```



# Our programming stack

- We mostly use garbage collected languages as
  - Go
  - Java
  - JavaScript



# Our programming stack

However, we are using  
non-garbage collected  
languages as **Rust** when  
needed



# Our friends for $\mu$ services



# We ❤️ open-source

## Code contribution:

- <https://github.com/ovh/beamium>
- <https://github.com/ovh/noderig>
- <https://github.com/ovh/tsl>
- <https://github.com/ovh/ovh-warp10-datasource>
- <https://github.com/ovh/ovh-tsl-datasource>
- ...

## Involved in:

- Warp10 community
- Apache Hbase/Flink development
- Prometheus/InfluxData discussions
- TS Query Language Working group

