Storage Performance Analysis Based on Kernel and Userspace Traces

Houssem Daoud

Progress Report meeting
May 2019

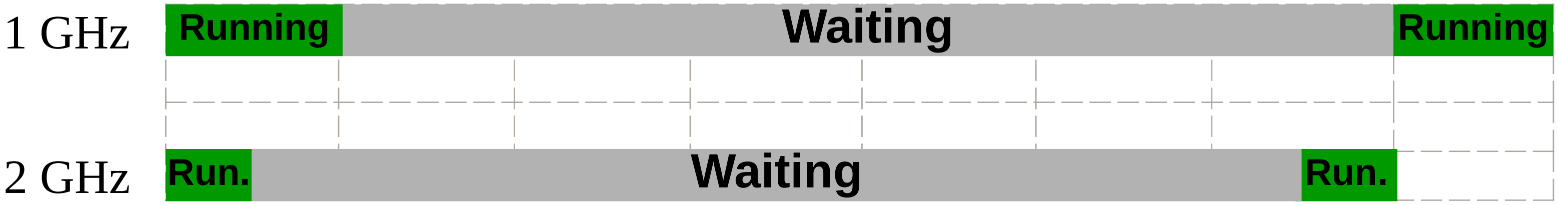
École Polytechnique de Montréal

Agenda

- **Introduction**
- Mass storage
  - Performance analysis of local storage devices
  - Performance analysis of distributed storage systems
- Main memory storage
  - Monitoring kernel memory usage
  - Performance analysis of automatic memory management mechanisms
- Conclusion and future work

Introduction

- Using faster processors doesn't always improve the performance of the system



A 2x faster processor gives an acceleration rate of 1.14x

- Storage operations are a major bottleneck in high-performance computing systems.
- Many mechanisms have been developed to improve the performance of storage operations (disk schedulers, memory allocators)

Introduction

- The complexity of those mechanisms makes them difficult to debug using traditional tools.

Benchmarking

- ➢ Synthetic workloads
- ➢ Doesn't help in finding the origin of the problem

Tracing

- ➢ Analyzes the behavior of real workloads
- ➢ Offers a more accurate insight into the internals of the storage subsystem

LTTng : a low overhead tracing framework

Introduction

- Tracing overhead can affect the normal behavior of the system (High frequency events).

- The amount of data generated by tracing is huge and needs to be post-processed

Objectives

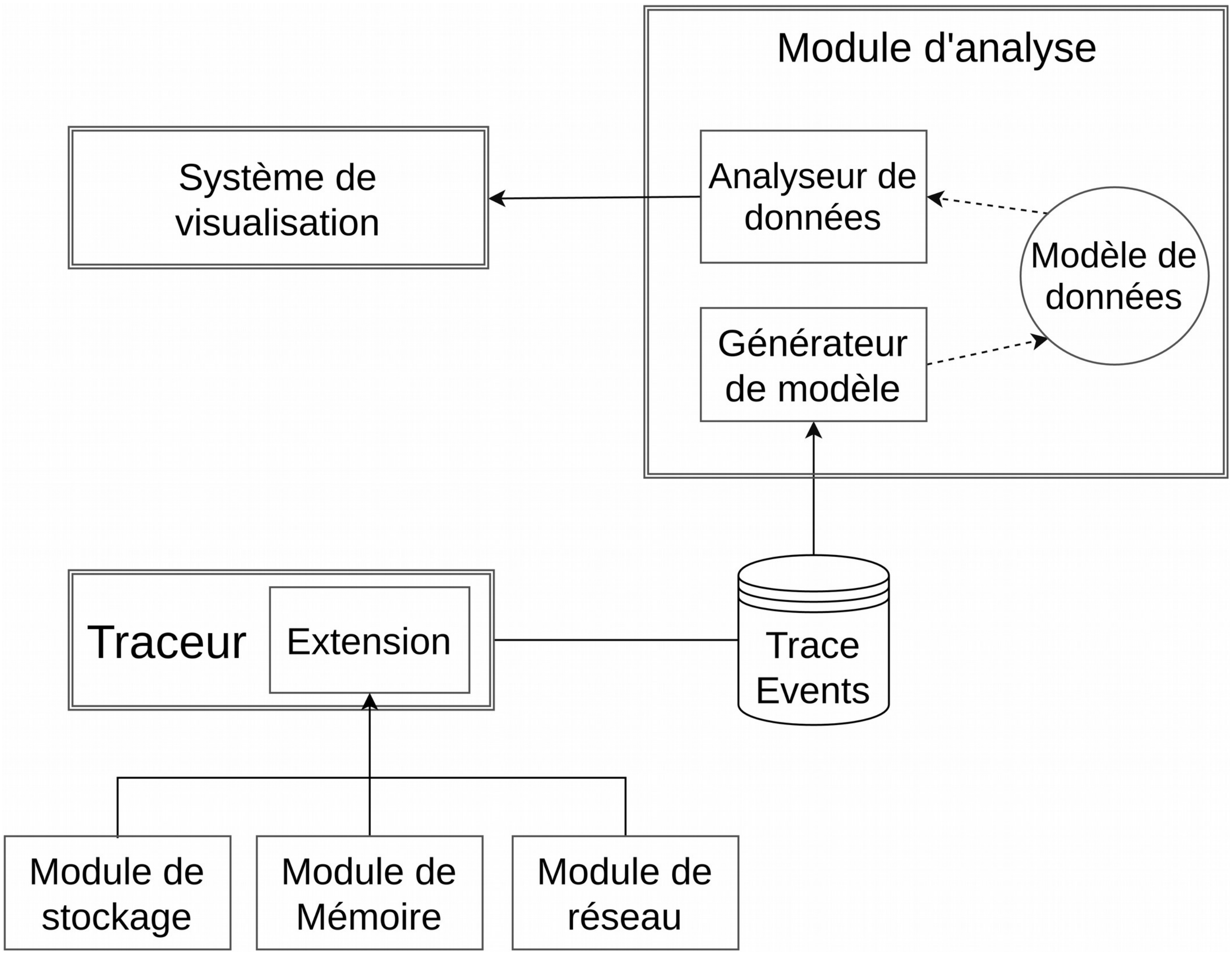
- Tracing the storage subsystem with a minimal overhead
- Analyzing the performance of the different storage systems
- Providing a comprehensive visualization system

Agenda

- Introduction
- **Mass storage**
  - **Performance analysis of local storage devices**
  - Performance analysis of distributed storage systems
- Main memory storage
  - Monitoring kernel memory usage
  - Performance analysis of automatic memory management mechanisms
- Conclusion and future work

Performance analysis of local storage devices

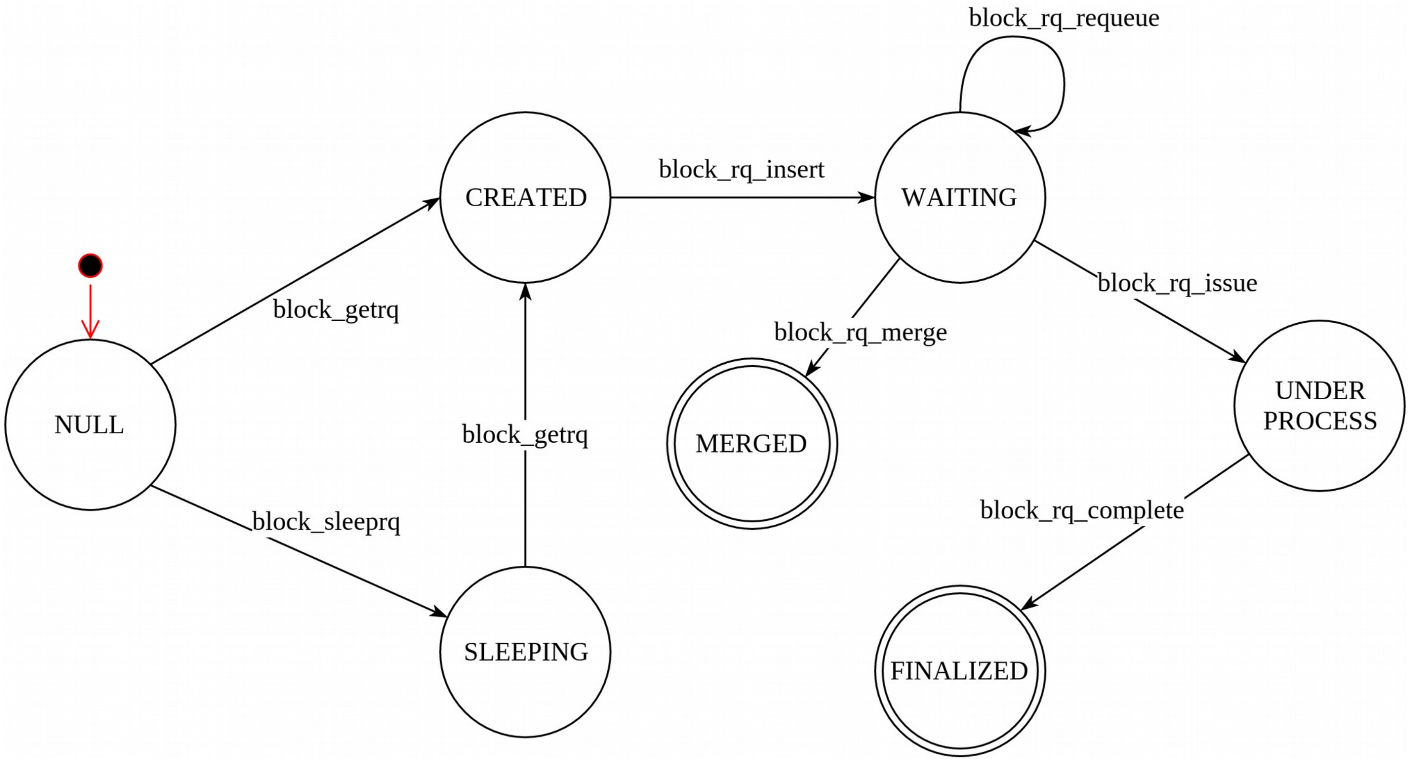
Proposed architecture



Performance analysis of local storage devices

Data Analysis

Stateful Analysis → The state of the system is kept in a historical database built incrementally in a single pass over the trace

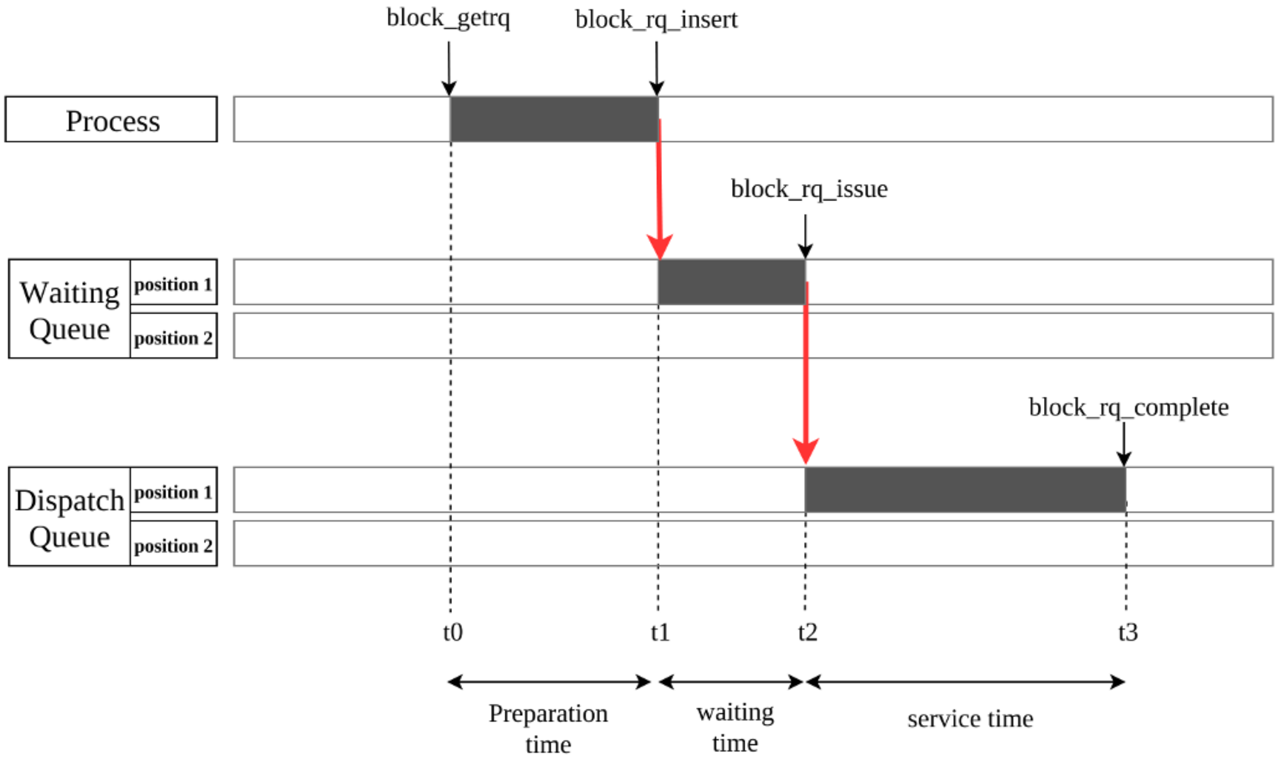


I/O request life cycle

Performance analysis of local storage devices

Metrics computation

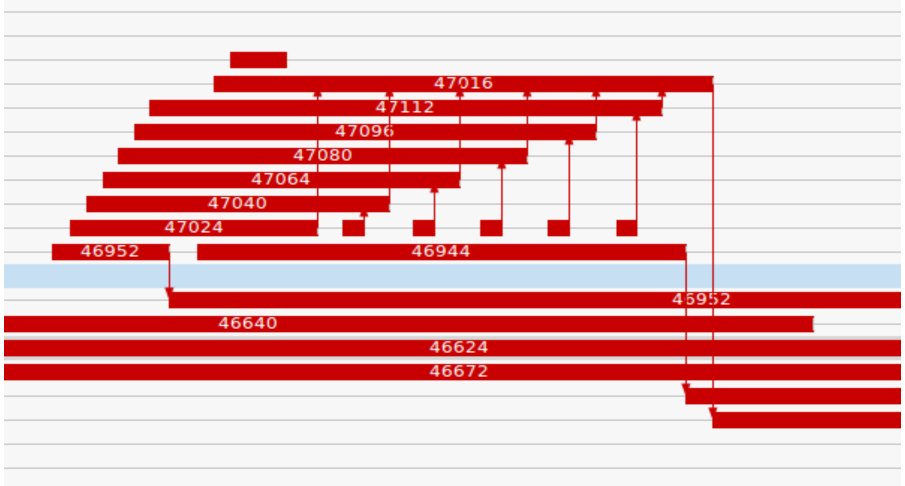
Latency



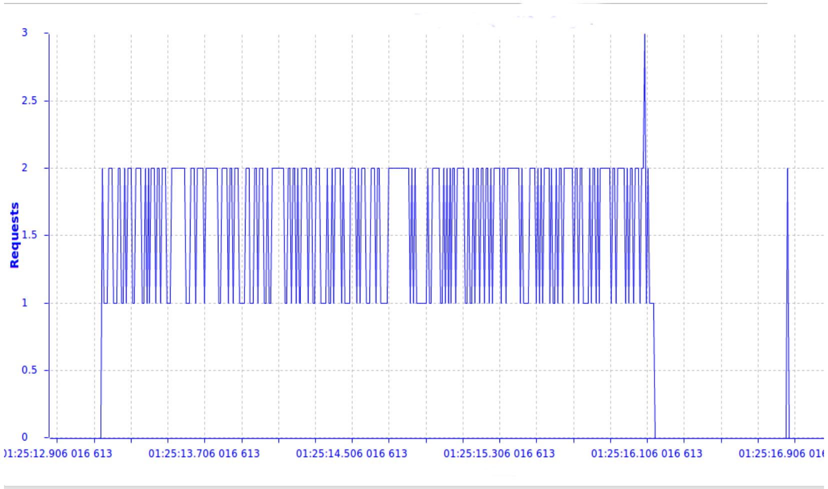
Latency = Preparation Time +Waiting Time + Service Time

Performance analysis of local storage devices

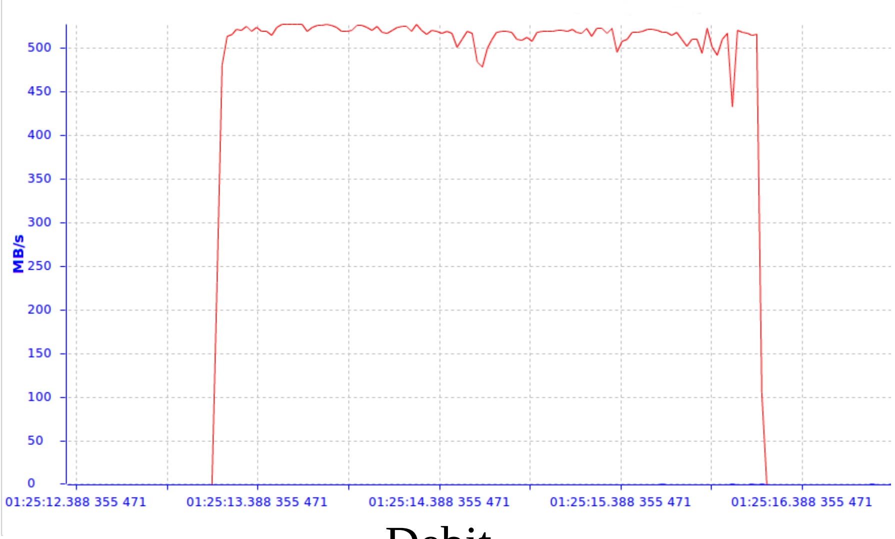
Visualization



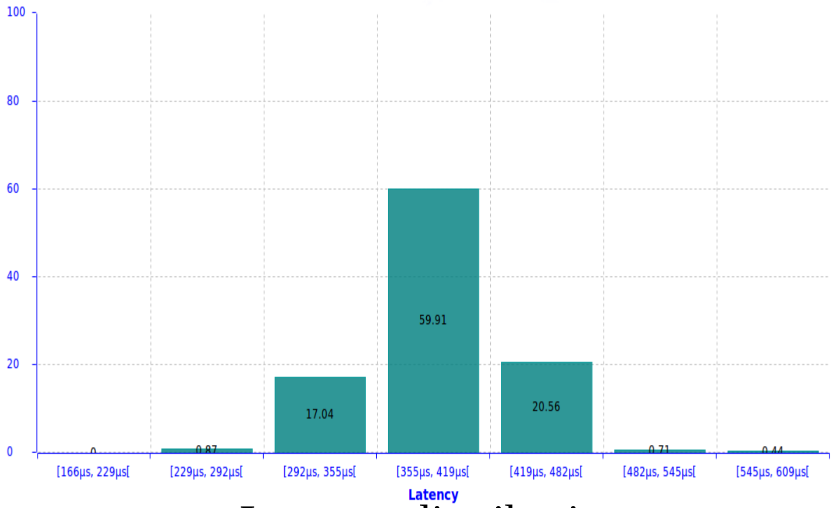
Disk waiting queues



Queue length



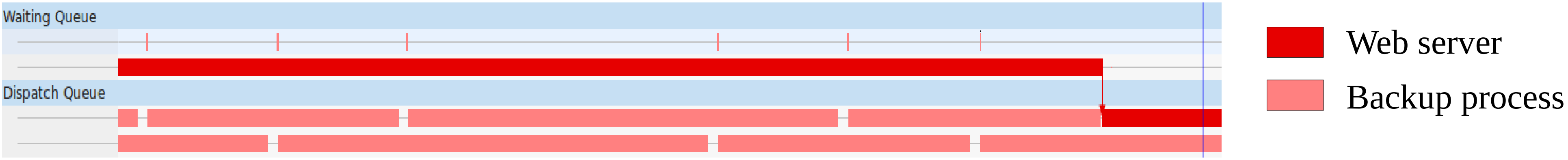
Debit



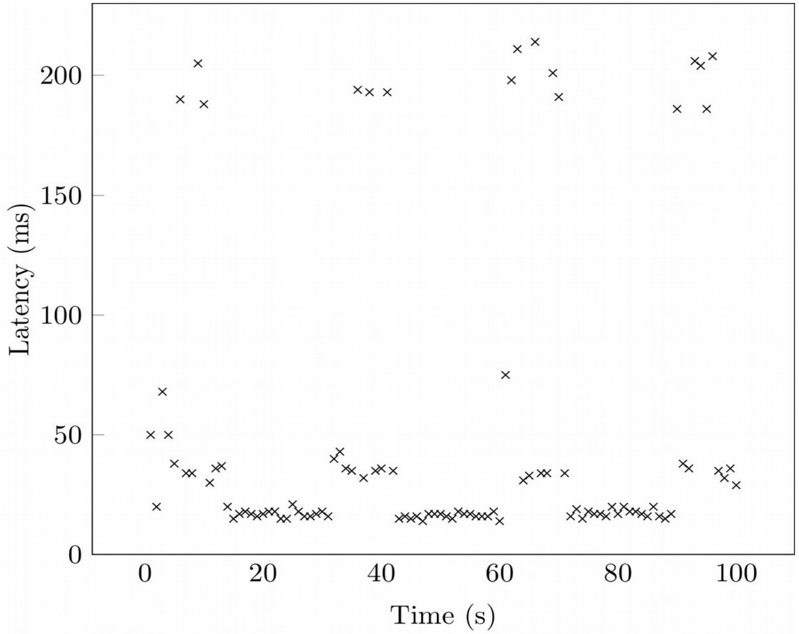
Latency distribution

Performance analysis of local storage devices

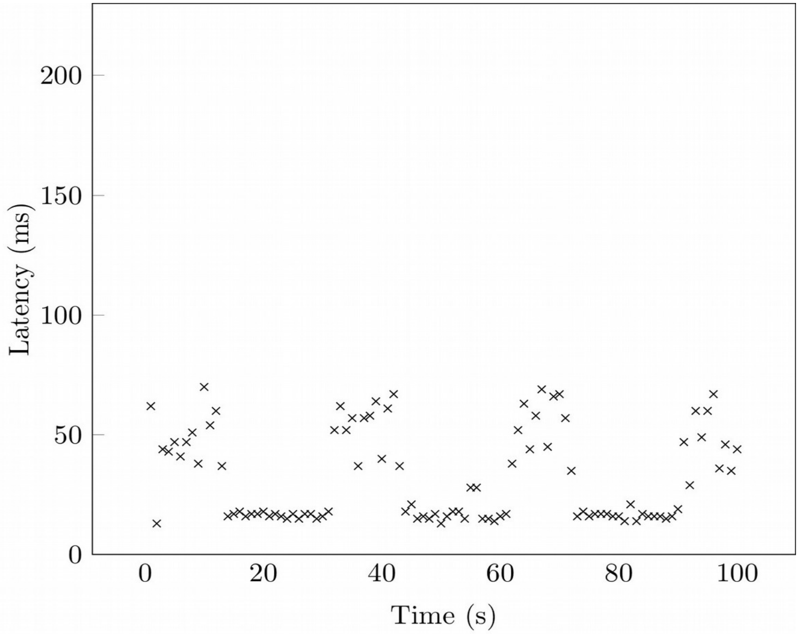
Use case: Investigating a high latency



Configuration problem : the backup process has a higher priority than the web server



Fixing the priority problem

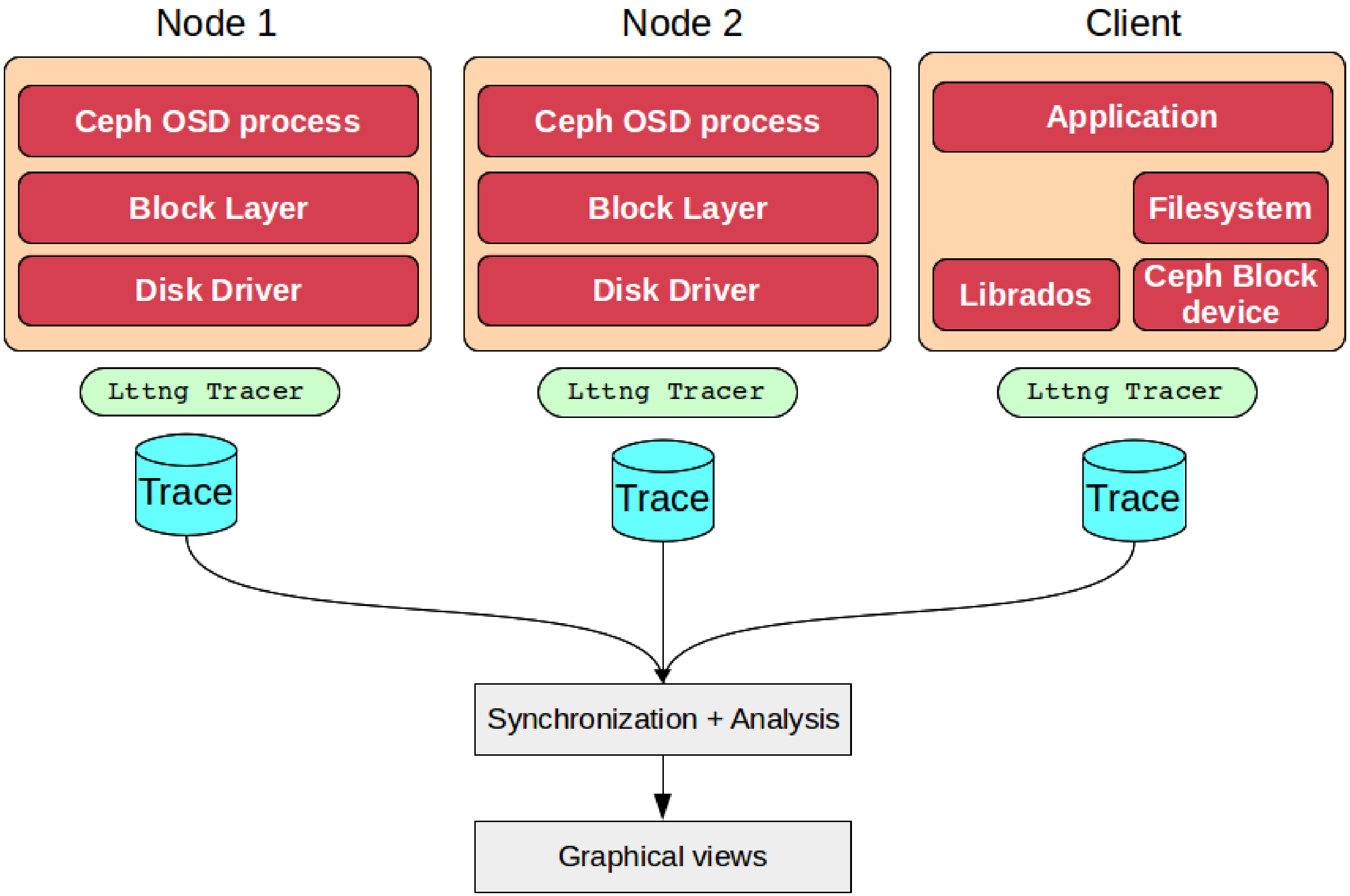


Agenda

- Introduction
- **Mass storage**
  - Performance analysis of local storage devices
  - **Performance analysis of distributed storage systems**
- Main memory storage
  - Monitoring kernel memory usage
  - Performance analysis of automatic memory management mechanisms
- Conclusion and future work

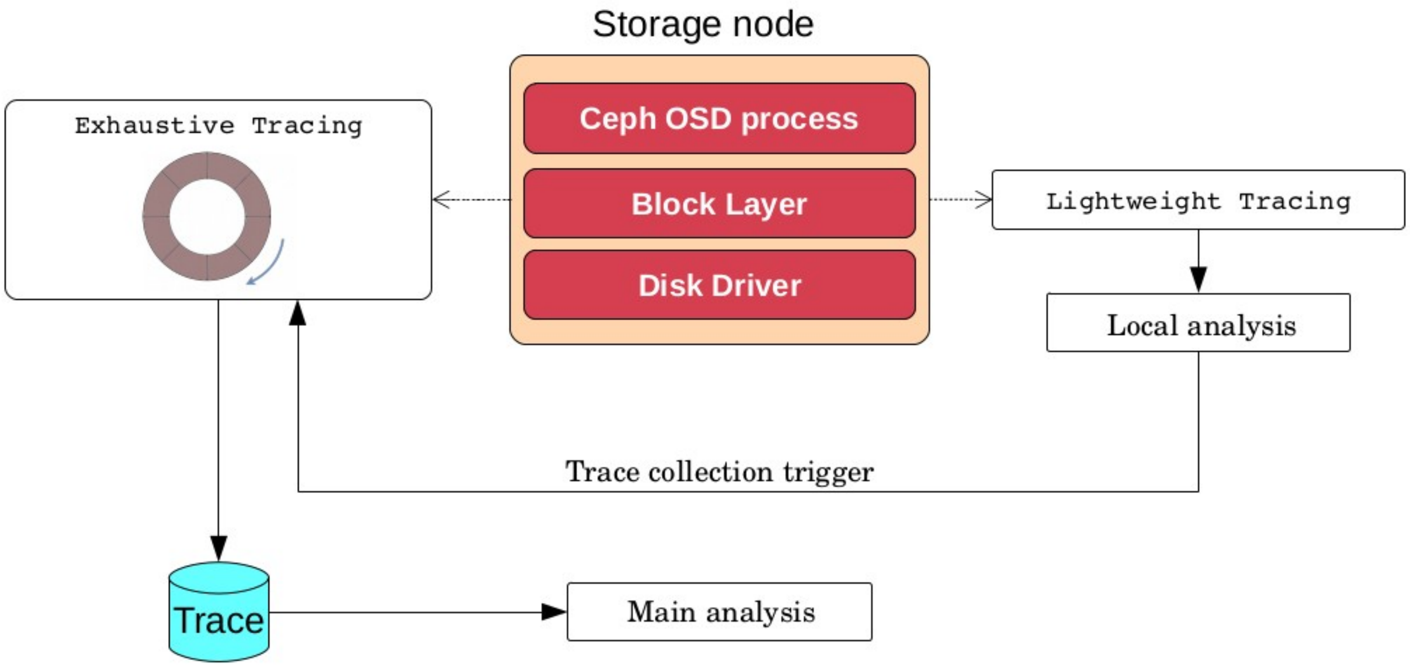
Performance analysis of distributed storage systems

Proposed architecture



Performance analysis of distributed storage systems

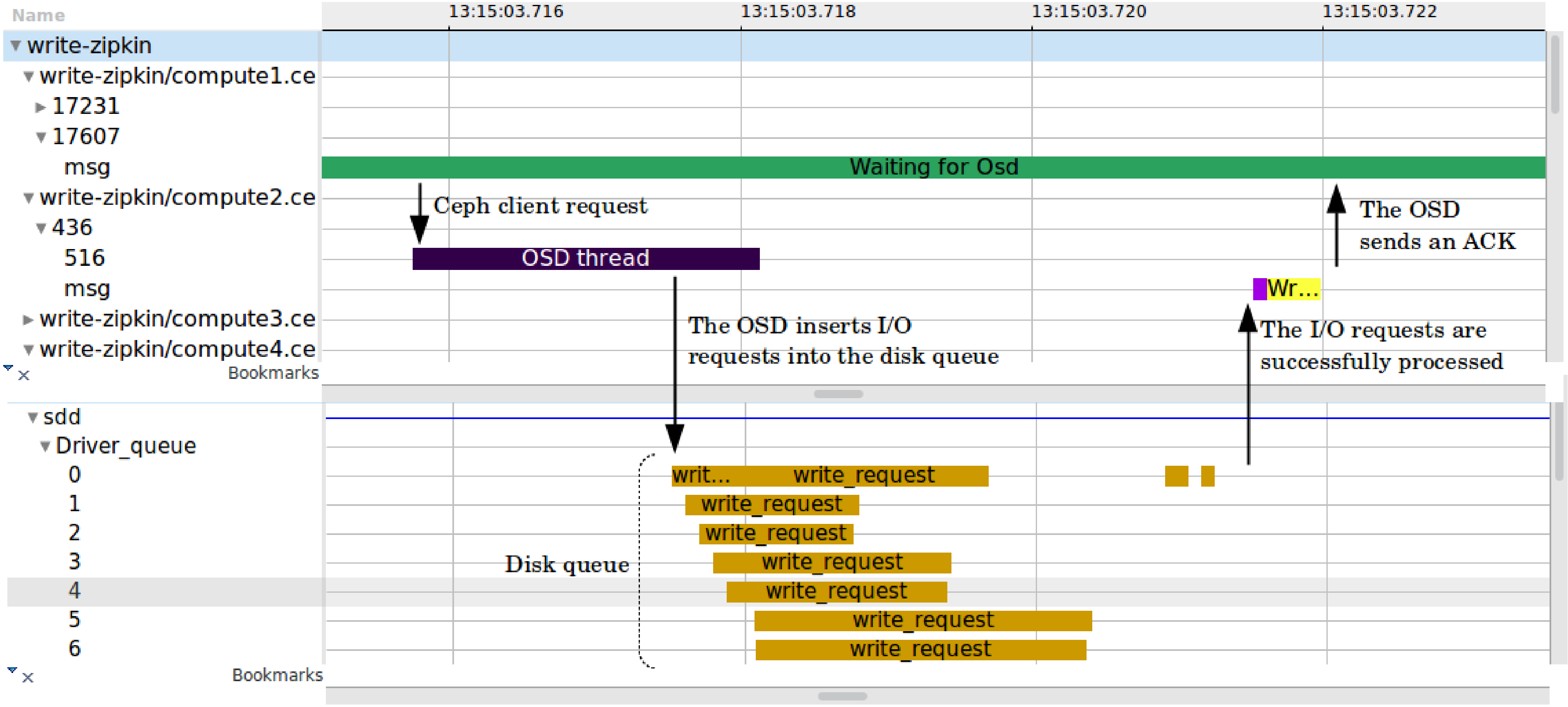
Data collection



- The lightweight tracing session traces a small number of events and analyzes them on the fly in order to to detect unusual behaviors
- The exhaustive tracing session writes the complete trace temporarily in a circular buffer
- The trace is only written if a problem is detected

Performance analysis of distributed storage systems

Visualization



Ceph client request

The OSD sends an ACK

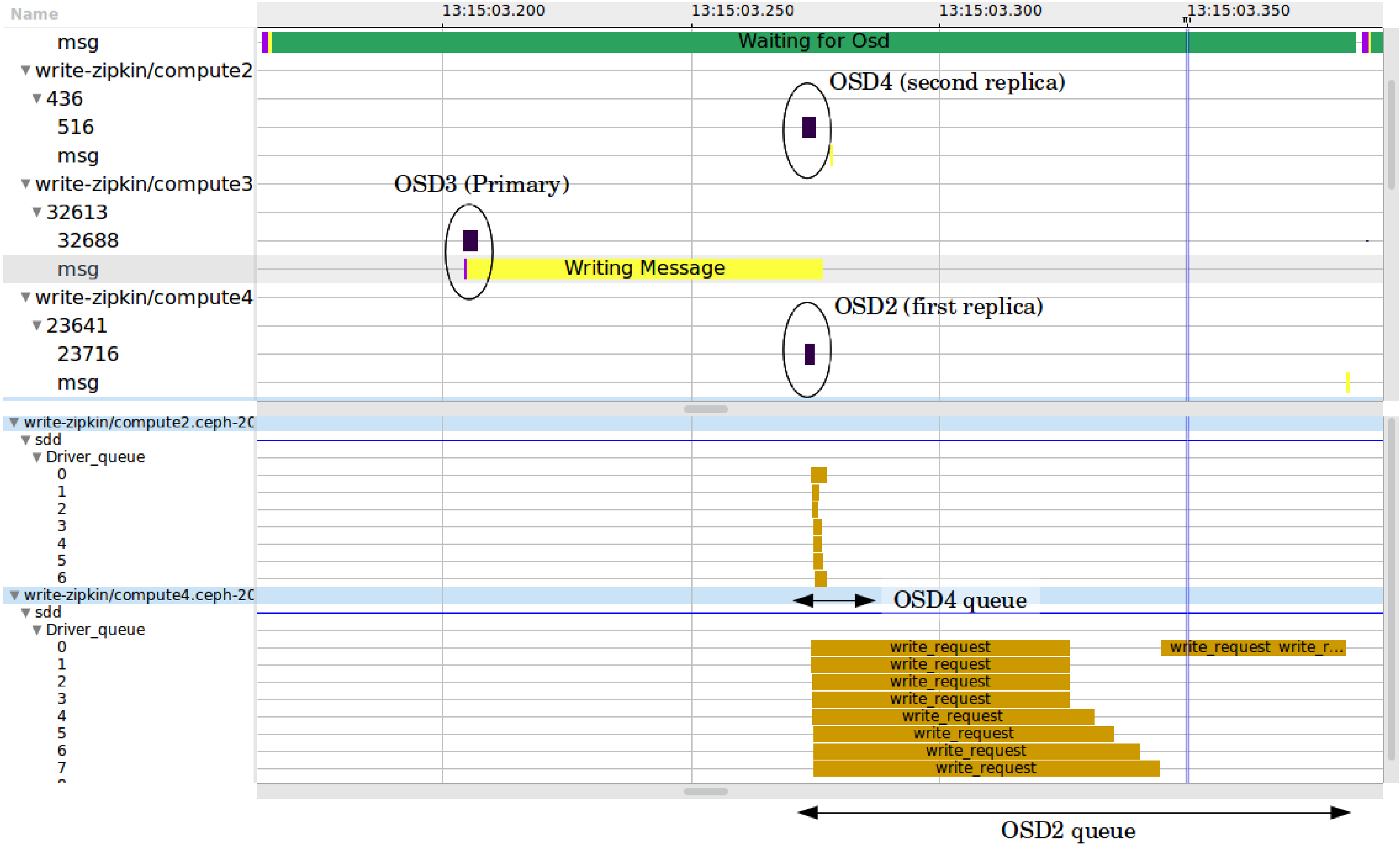
The OSD inserts I/O requests into the disk queue

The I/O requests are successfully processed

Disk queue

Performance analysis of distributed storage systems

Use case: Impact of a slow disk



The client has to wait until the replication is successfully completed in all the secondary OSDs.

Agenda

- Introduction
- Mass storage
  - Performance analysis of local storage devices
  - Performance analysis of distributed storage systems
- **Main memory storage**
  - **Monitoring kernel memory usage**
  - Performance analysis of automatic memory management mechanisms
- Conclusion and future work

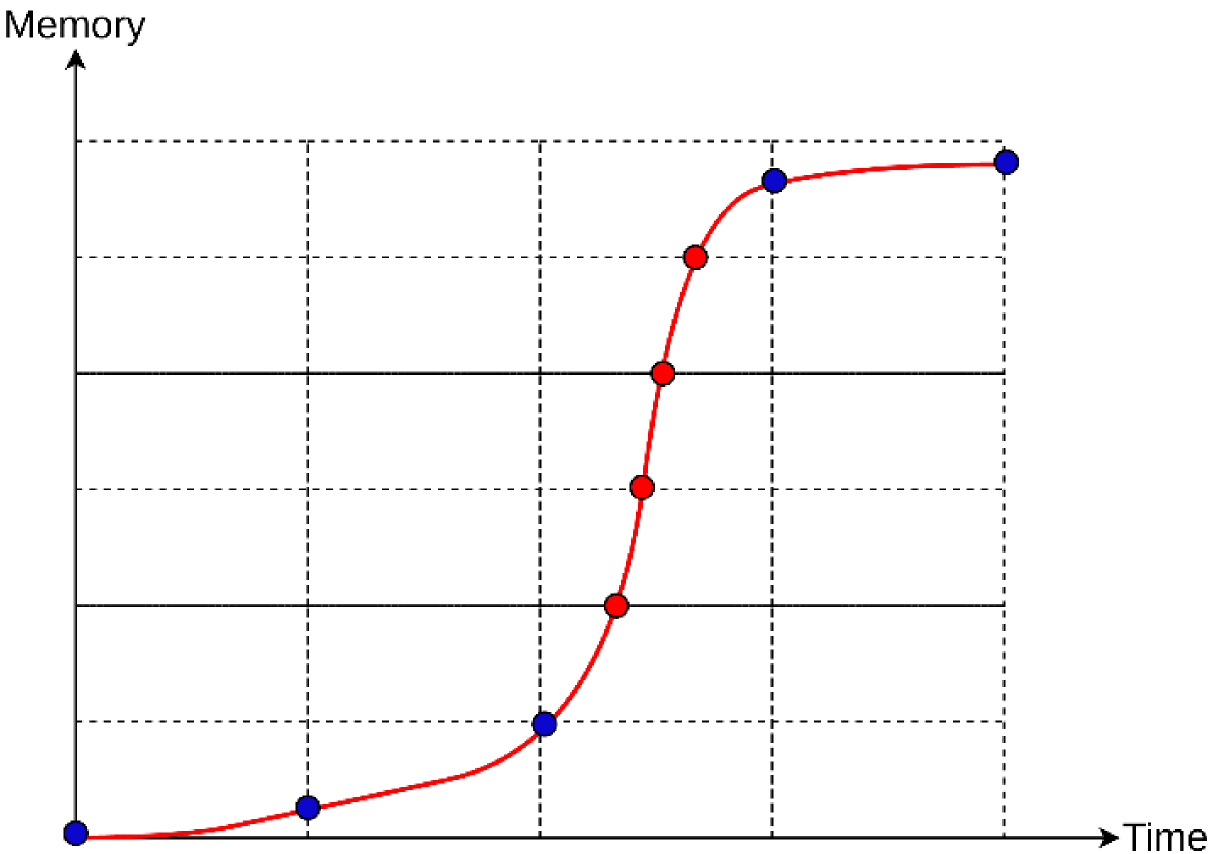
Monitoring kernel memory usage

Dynamic Trace-based Sampling Algorithm

- An event is triggered if memory variability exceeds a certain threshold

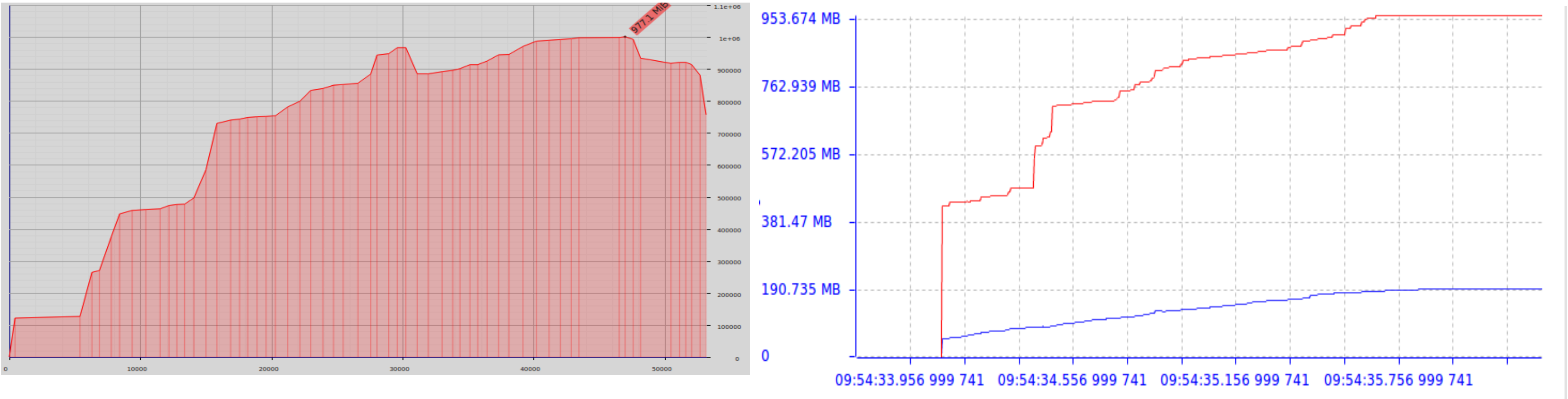
- Implemented as a Kernel module.

- Lock-free data structures are used to provide a good scalability

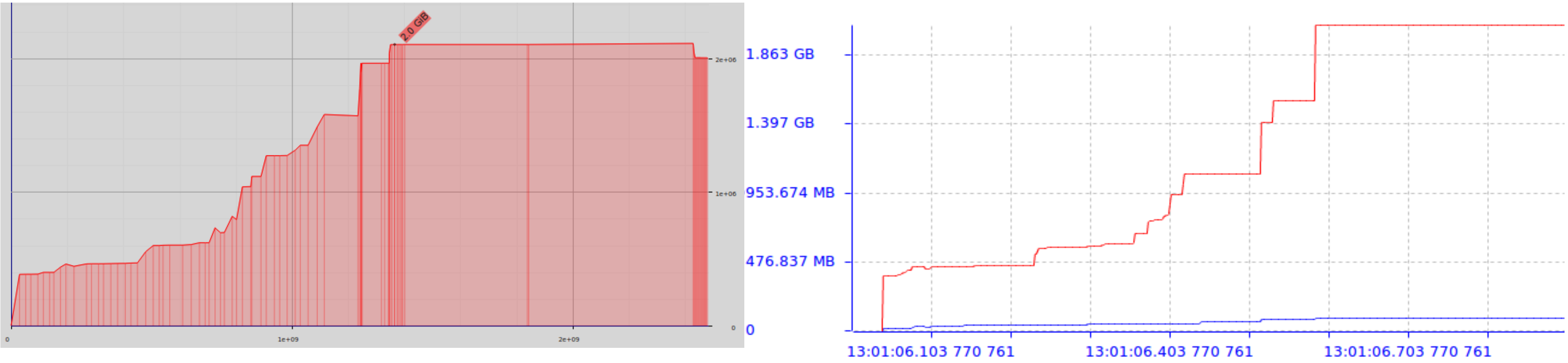


Monitoring kernel memory usage

Use cases



Firefox memory usage at startup using Massif / Dynamic Sampling Algorithm

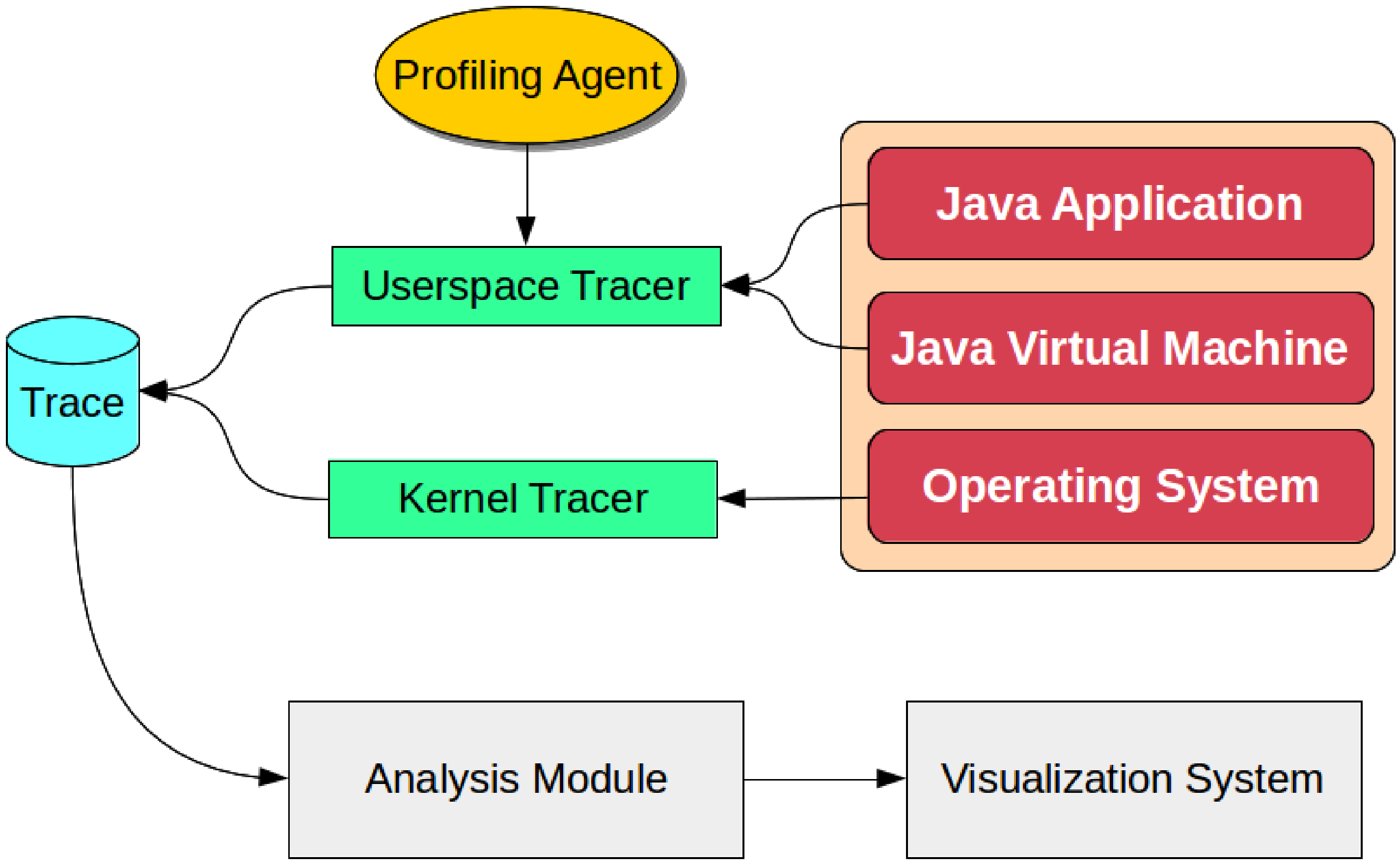


Totem video player memory usage using Massif / Dynamic Sampling Algorithm

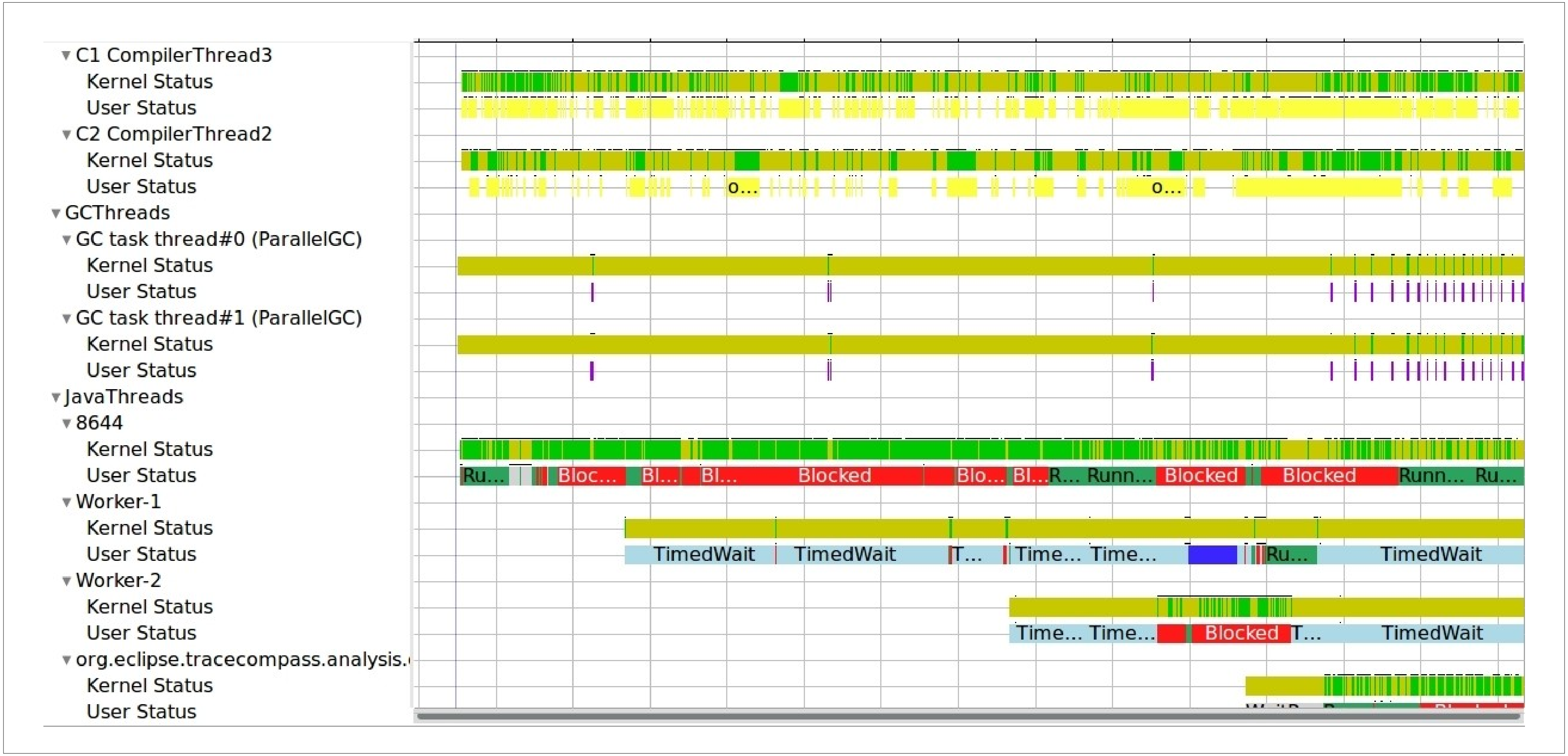
Agenda

- Introduction
- Mass storage
  - Performance analysis of local storage devices
  - Performance analysis of distributed storage systems
- **Main memory storage**
  - Monitoring kernel memory usage
  - **Performance analysis of automatic memory management mechanisms**
- Conclusion and future work

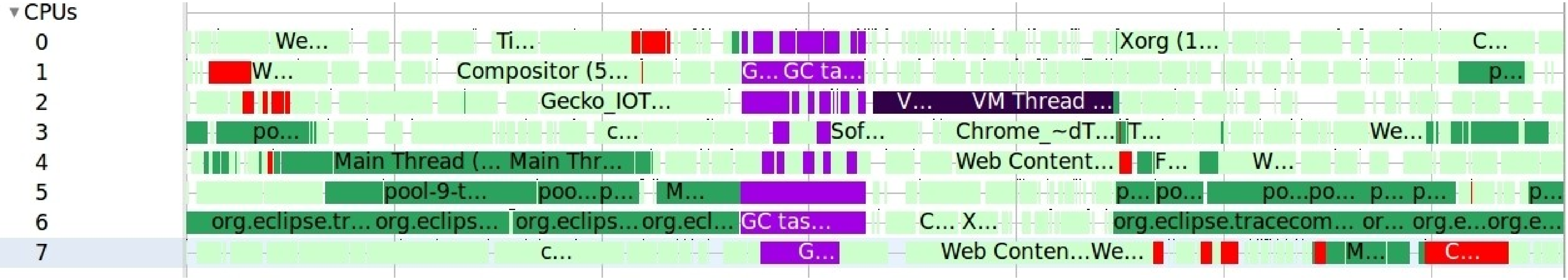
Proposed architecture



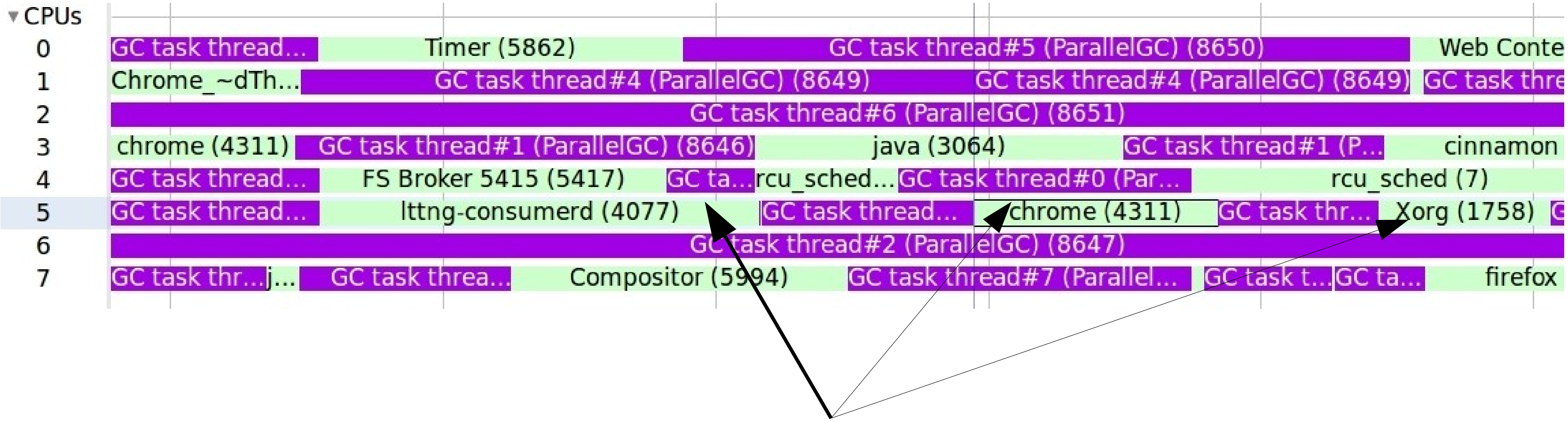
Performance analysis of automatic memory management mechanisms



Performance analysis of automatic memory management mechanisms


CPUs
0 We... Ti... Xorg (1... C...
1 W... Compositor (5... G... GC ta... p...
2 Gecko_IOT... V... VM Thread ...
3 po... c... Sof... Chrome_~dT... T... We...
4 Main Thread (... Main Thr... Web Content... F... W...
5 pool-9-t... poo... p... M... p... po... po... po... p... p... p...
6 org.eclipse.tr... org.eclips... org.eclips... org.ecl... GC tas... C... X... org.eclipse.tracecom... or... org.e... org.e...
7 c... G... Web Conten... We... M... C...

This view shows on which CPU each Java thread is running


CPUs
0 GC task thread... Timer (5862) GC task thread#5 (ParallelGC) (8650) Web Conte
1 Chrome_~dTh... GC task thread#4 (ParallelGC) (8649) GC task thread#4 (ParallelGC) (8649) GC task thre
2 GC task thread#6 (ParallelGC) (8651)
3 chrome (4311) GC task thread#1 (ParallelGC) (8646) java (3064) GC task thread#1 (P... cinnamon
4 GC task thread... FS Broker 5415 (5417) GC ta... rcu_sched... GC task thread#0 (Par... rcu_sched (7)
5 GC task thread... lttng-consumerd (4077) GC task thread... chrome (4311) GC task thr... Xorg (1758) G
6 GC task thread#2 (ParallelGC) (8647)
7 GC task thr... j... GC task threa... Compositor (5994) GC task thread#7 (Parallel... GC task t... GC ta... firefox

The view shows that a GC thread is being preempted by other processes

Agenda

- Introduction
- Mass storage
  - Performance analysis of local storage devices
  - Performance analysis of distributed storage systems
- Main memory storage
  - Monitoring kernel memory usage
  - Performance analysis of automatic memory management mechanisms
- **Conclusion and future work**

Conclusion

- Recovering Disk Storage Metrics from Low-level Trace Events (journal - published)
- Performance Analysis of Distributed Storage Clusters Based on Kernel and Userspace Traces (journal - submitted)
- Dynamic Trace-based Sampling Algorithm for Memory Usage Tracking of Enterprise Applications (conference - published)
- Multilevel Analysis of The Java Virtual Machine Based on Kernel and Userspace Traces (journal - submitted)

Future work

- Live tracing support
- Using machine learning algorithms to detect and classify performance problems