Cloud Computing Applications and Services

(Aplicações e Serviços de Computação em Nuvem)

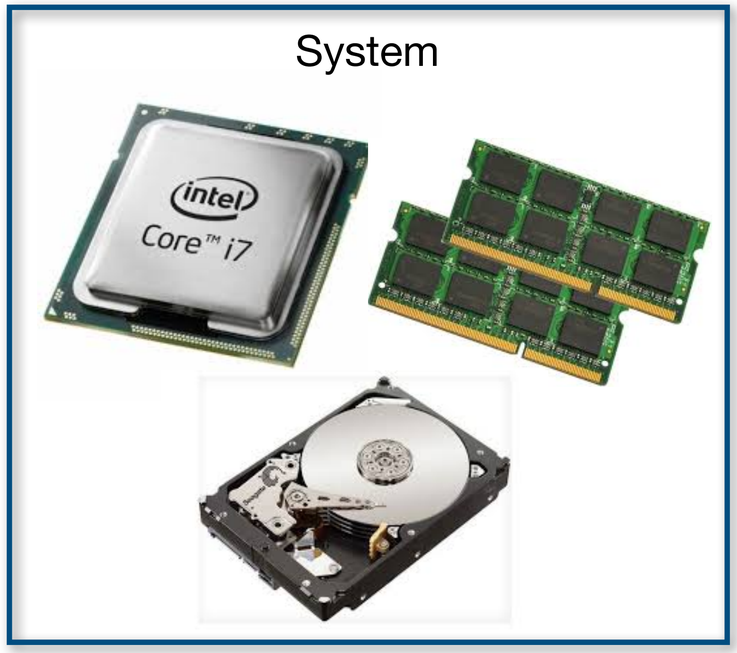
Monitoring

University of Minho

2024-2025

Monitoring
Definition of monitor

- A **monitor** observes the **activity** of a system

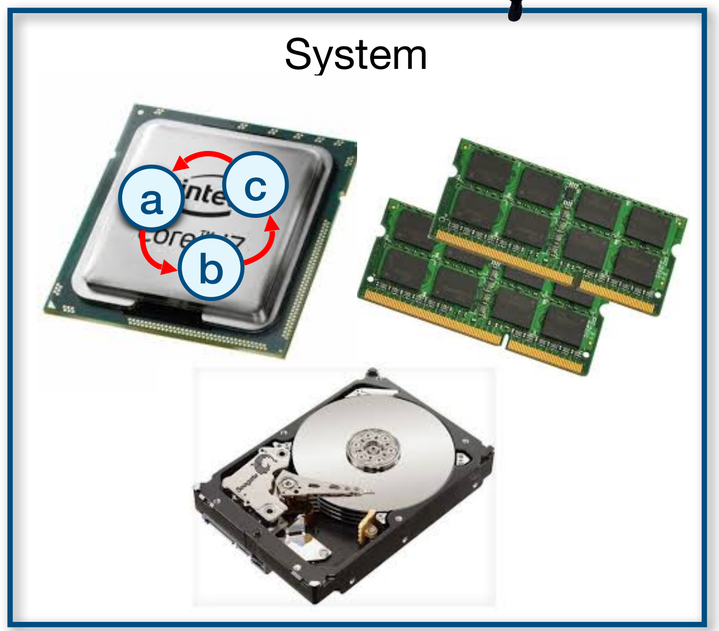


Monitor

Monitoring concepts

State, events and traces

- A system contains physical and logical resources with **state**
  - CPU, RAM, Disk, virtual memory, processes, threads, …
- **State** changes as **events**
  - CPU / memory instructions, I/O operations, application requests…
- A **trace** is a log of events
  - Each entry at the log typically contains a **timestamp** and **other details of the event** (e.g., type, duration, args, …)



Trace

10:00 – details of events a
10:01 – details of event b
...

Monitoring concepts

Domain and detail

- The **domain** is the set of activities (events) observed
  - E.g., resource consumption (CPU, RAM, network, disk), I/O operations, database requests
- The **detail** of the information being captured (monitored) varies:
  - According to the **input rate**
    - How many events can the monitor observe per unit of time?
  - According to the **resolution**
    - At what granularities (e.g., ms, us, ns resolution) can the monitor observe events?
- A monitor imposes **overhead**, changing the observed activity
  - E.g., the monitor also consumes system resources (e.g., CPU, RAM, I/O), such may interfere with the system being monitored and the observations

Monitor classification

Some main features

- What triggers the observation/collection of events?
  - **Event-driven:** Observation is triggered every time an event of interest occurs
  - **Sampling:** Observation is triggered only for a subset of events (e.g., every 100 events, or every 10 ms…)
  - One is **trading accuracy** (observing as many events as possible - *event-driven*) for **performance** (less overhead introduced by the monitor - *sampling*)
- When are the observations available for users and/or systems?
  - **On-line:** Collected events are sent straight away to the user and/or system (e.g., to a file or a database)
  - **Batch:** Events are grouped and sent in a batch to the user and/or system
  - One is **trading real-time observation** (on-line) for **better performance** (batch)

Monitor classification

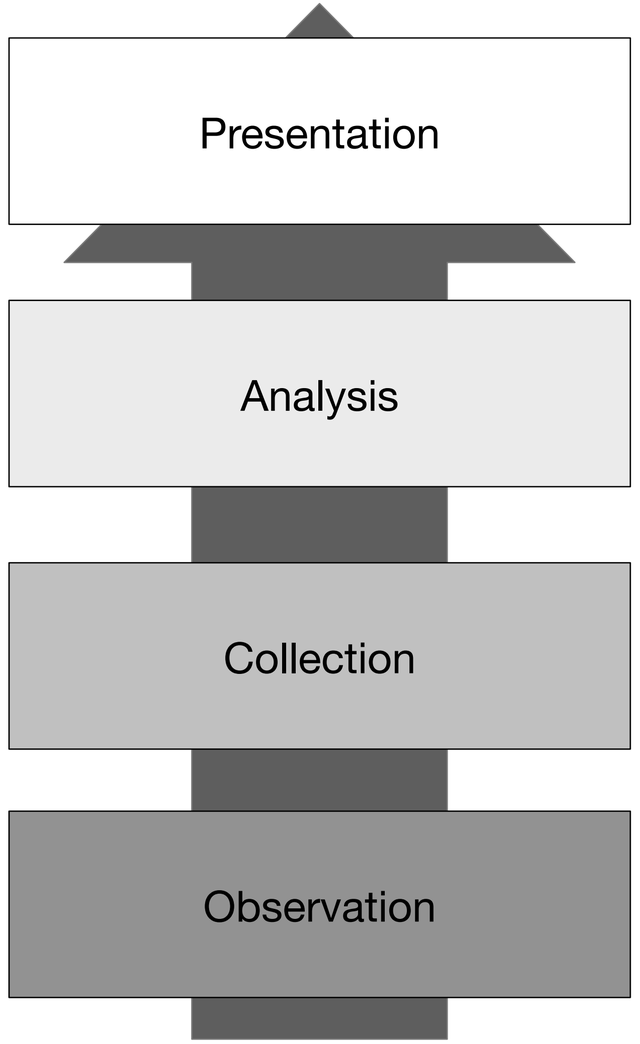
Some main features

- How is the monitor implemented?
  - **Hardware** monitors are *typically* more **accurate** and have better **resolution**
  - **Software-based** monitors are usually **less expensive**, as well as **more flexible** and **extensible** in terms of the metrics and events one can observe
- How is the monitor designed distribution-wise?
  - **Centralized:** the monitor service is deployed on a single node (e.g., server)
  - **Distributed:** the service is spread, and even replicated, across several nodes
- What about the scope of monitoring distribution-wise?
  - **Single node:** events collected for a single node
  - **Distributed:** events collected for a cluster of nodes
  - This may be orthogonal to the distributed or centralized design of the monitor service. Do not confuse the **design of the monitor** with its **distribution scope**!

Monitor architecture

Main layers

- **Observation** of raw events in systems
  (e.g., resource usage, I/O requests, …)
- **Collection** and normalization of data
  (e.g., normalization of time units)
- **Analysis** of normalized data
  (e.g., filter, querying, summarizing)
- **Presentation** of the analysis results
  (e.g., dashboards, reports, alarms)

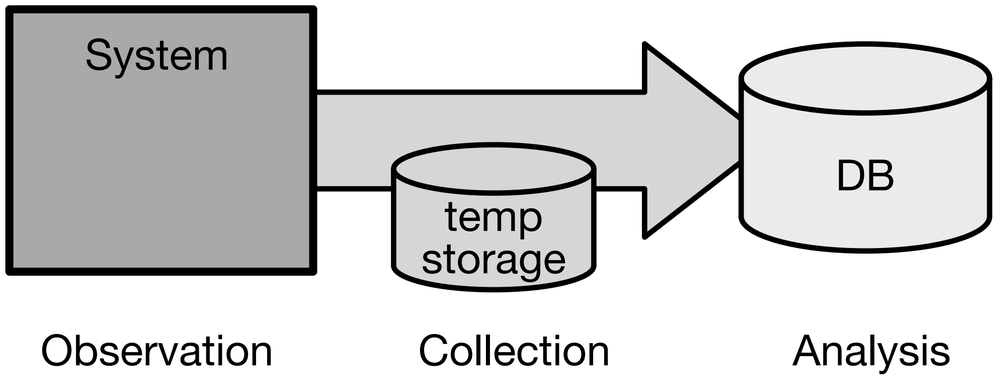


Observation Layer

- Events of interest can be observed through different techniques
- **Passive observation or spying:** no need to instrument the resources and/or applications being monitored. The monitor passively observes the system
  - E.g., network sniffer
- **Instrumentation:** The hardware, operating system, and/or applications are instrumented (e.g., code is changed) to observe relevant information
  - E.g., hardware counters, strace, eBPFs, …
- **Probing:** The monitor interacts (probes) the system / application being observed to collect metrics
  - E.g., with the ping command

Collection Layer

- Observed events can be collected and normalized through two approaches
  - **Push-based:** events are sent by the observation layer to the collection one
  - **Pull-based:** events are pulled by the collection layer from the observation one
- Besides collecting data, the layer is used to **normalize** it
  - E.g., aggregate data, normalize time intervals, synchronize time when events are collected across multiple nodes
- In some cases, the layer may provide **persistency** (temporary storage) for collected events



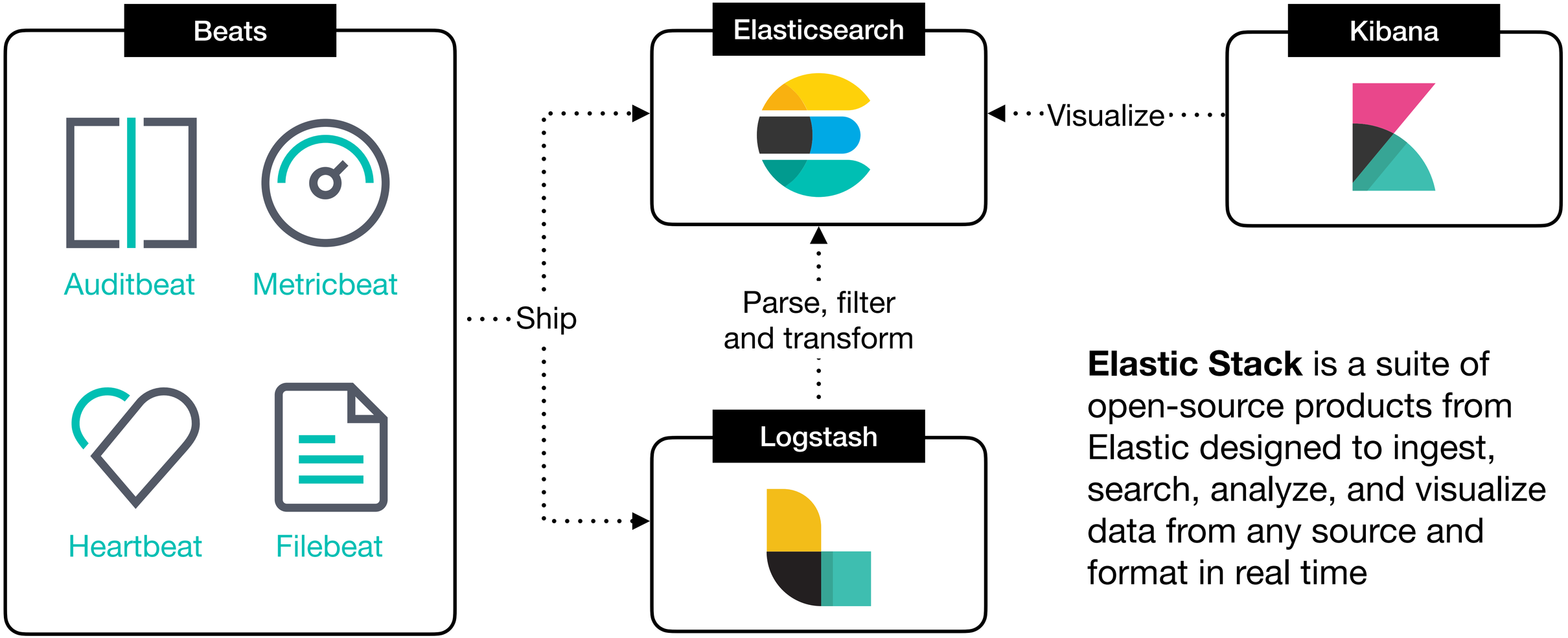
Analysis Layer

- The analysis layer has two main goals
- Efficient and reliable **data storage and indexing**
  - If a comprehensive set of events is collected, the analysis component may have to persist and index large volumes of data
- Efficient **data processing**
  - Stored and indexed data must be queryable, and sometimes analyzed in streams (e.g., for time-series analysis)
- The choice of technology for this layer depends on the type of data being analyzed and the analysis requirements
  - E.g., time-series database, document-oriented database, …

Presentation Layer

- The presentation layer can be used for visualizing
  - **Performance metrics** of applications and/or systems (e.g., requests throughput and latency, CPU usage, RAM consumption, …)
  - **The configuration** of cluster resources and the applications deployed in these
  - **Errors** at cluster resources and/or applications
- Users can use this layer through different visual representations
  - E.g., dashboards, reports, alarms, …

elastic stack



Elastic Stack is a suite of open-source products from Elastic designed to ingest, search, analyze, and visualize data from any source and format in real time

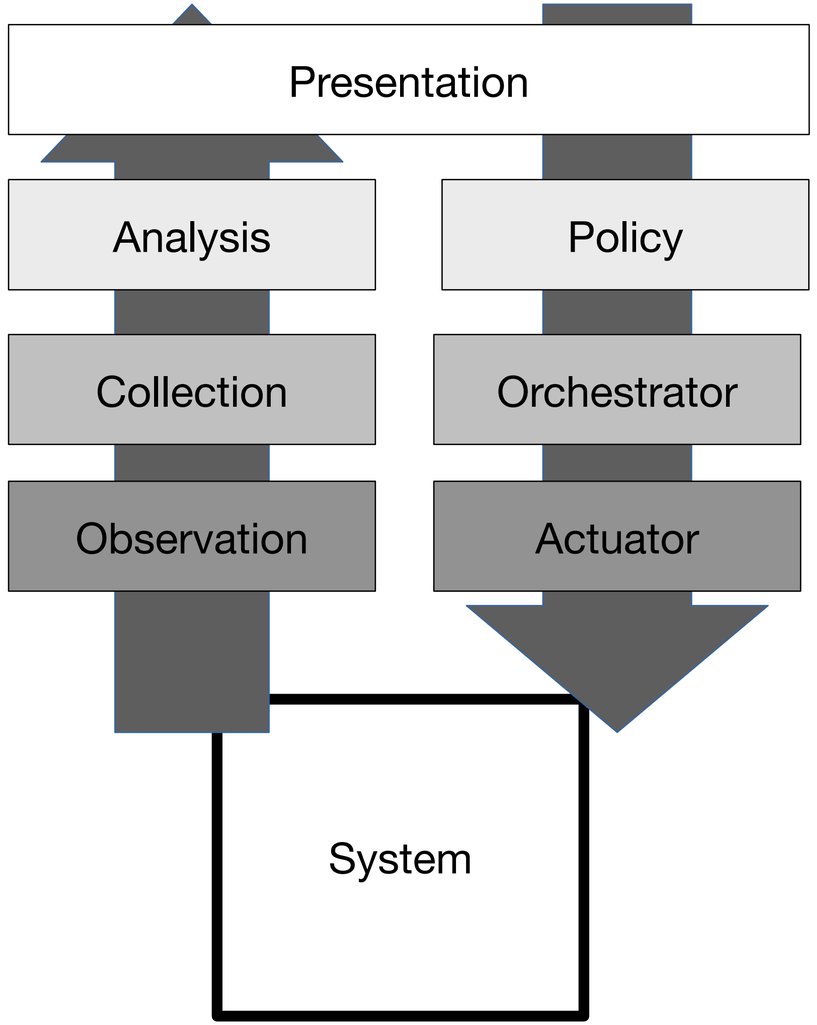
Elastic Stack

- **Beats:** lightweight data shippers (Observation and Collection)
  - *Purpose: t*o observe, collect and send data to Logstash or Elasticsearch for further processing and indexing
- **Logstash:** a data ingestion tool (Collection)
  - *Purpose:* to collect, transform, and send data from various sources to Elasticsearch
- **Elasticsearch:** a distributed, JSON-based search and analytics engine (Analysis)
  - *Purpose:* to index, search, and analyze data
- **Kibana:** a data visualization and exploration tool (Presentation)
  - *Purpose:* to visualize and manage data from Elasticsearch

Find more about Elastic Stack at: https://www.elastic.co/elastic-stack

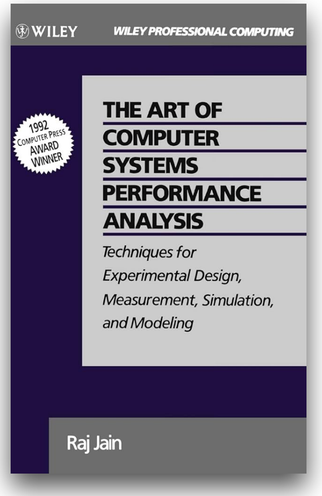
Monitoring and Actuation

- Monitoring can be combined with actuation to automatically apply actions at the system
- The actions to apply are specified as **policies**
  - E.g., when a server's CPU is above a given threshold migrate some VMs to another server
- The **orchestrator** is responsible for defining the best strategy for applying policies
  - E.g., choose what VM(s) to migrate and to which server(s)
- The **actuator** is responsible for applying the strategy defined by the orchestrator
  - E.g., migrate the VM resource to another server



Further Reading

- R. Jain, ***"The Art of Computer Systems Performance Analysis."*** Wiley, 1991.
  - Chapters 7 and 8
- Esteves T, Neves F, Oliveira R, Paulo J. 2021. ***CaT: Content-aware Tracing and Analysis for Distributed Systems***. ACM/IFIP Middleware conference (Middleware).
- Esteves T, Macedo R, Oliveira R, Paulo J. 2023. ***Toward a practical and timely diagnosis of applications' I/O behavior***. IEEE Access journal.



Questions?