# 17th feb: Monitoring and Alerting

**Contents:**

What is Monitoring?  
What do we Use Monitoring?  
Types of Monitoring  
How do we Moniter  
Monitoring Alerts?  
where is monitoring is Performed?  
Best Practices for monitoring   
Tools for monitering  
Reports and matrices  
 SLA and SLO

**What is Monitoring?**  
**Monitoring** is the process of continuously observing and analyzing the performance, availability, and health of systems, applications, networks, or infrastructure. It helps identify issues, improve performance, and ensure stability.  
  
**Why Do We Use Monitoring?**  
**Importance of Monitoring:**

* **Proactive Issue Detection**: Identifies potential problems before they escalate.
* **Performance Optimization**: Ensures applications and systems run efficiently.
* **Security Assurance**: Detects anomalies and potential security threats.
* **Compliance & Reporting**: Helps meet regulatory and operational compliance.
* **Improved User Experience**: Ensures service availability and reliability.

**Types of Monitoring**  
  
1. Infrastructure Monitoring:  
- Monitors physical and virtual infrastructure, including servers, storage, and network devices.  
- CPU usage, memory, disk I/O, network traffic, uptime.  
- Tools: Prometheus, Nagios, Zabbix, Datadog, SolarWinds.  
  
2. Application Performance Monitoring (APM):  
- Tracks application performance, response times, and user experience  
- Latency, throughput, error rates, API response time.  
- Tools: New Relic, Dynatrace, AppDynamics, Splunk APM.  
  
3. Network Monitoring:  
- Observes network traffic, latency, bandwidth, and connectivity issues.

* Packet loss, latency, bandwidth utilization.

**- Tools:** Wireshark, PRTG Network Monitor, SolarWinds

4. Database Monitoring:  
- Ensures database performance, query efficiency, and availability.  
- Query execution time, connection errors, indexing issues  
- **Tools:** MySQL Enterprise Monitor, Percona Monitoring,   
 pgAdmin, DataDog.

**3. How Do We Monitor?**

### **Setting Up Alerts Based on Criticality**

* **SEV1 (Critical)**: System-wide failure affecting business operations.
* **SEV2 (High)**: Major functionality impacted with limited workarounds.
* **SEV3 (Medium)**: Performance degradation but service remains functional.
* **SEV4 (Low)**: Minor issues or informative alerts.

### **Defining Monitoring Strategies Across Environments**

* **Development (Dev)**: Logging and debugging tools.
* **Testing**: Load testing, functional monitoring.
* **Pre-Production (Pre-Prod)**: Simulation of production conditions.
* **Production (Prod)**: Real-time monitoring, alerting, auto-scaling.

### **Tools Used for Monitoring**

* **Prometheus**: Open-source monitoring and alerting tool.
* **Grafana**: Visualization and dashboarding.
* **Graylog**: Log management and analysis.

**Observability Vs Monitoring:**

|  |  |  |
| --- | --- | --- |
| Terms | Observability | Monitoring |
| Define | Provides deep insights into system behavior, even for unknown issues. | Tracks system health using predefined metrics. |
| Focus | Understanding **why something went wrong**. | Detecting **known problems**. |
| Data source use | Logs, metrics, traces, distributed systems data. | Logs, metrics, alerts. |
| When to use | For **troubleshooting complex issues** and debugging. | For **basic health checks** and performance monitoring. |

**Incident Response and alert Handling:**  
When a system fails or behaves abnormally, quick **incident response and alert handling** are crucial to **minimize downtime, reduce impact, and restore services** efficiently  
  
**1. What is an Incident?**

An **incident** is any **unexpected issue** that disrupts the normal functioning of a system, application, or network.

### **Examples of Incidents:**

* A website going **down** (HTTP 500 errors).
* High **CPU usage** causing slow performance.
* A **database crash**, making the application unresponsive.

## **Incident Response Lifecycle**

The **Incident Response Lifecycle** follows a structured approach to detecting, analyzing, and resolving incidents.

### **Step 1: Detection & Alerting**

* **Monitoring tools** (e.g., Prometheus, CloudWatch, Nagios) continuously track system health.
* If an anomaly is detected (e.g., high latency, server crash), an **alert** is triggered.
* Alerts are **classified** based on severity (SEV1 to SEV4).

### **Step 2: Triage & Assessment**

* Engineers analyze **logs, metrics, and traces** to understand the issue.
* **Key questions to ask:**
  + What systems are affected?
  + What is the root cause?
  + How many users are impacted?
  + Has this happened before?

### **Step 3: Containment & Mitigation**

* Apply **quick fixes** to **limit the damage** while investigating the root cause.
  + Example: Restarting a **crashed database** or rolling back a faulty **deployment**.
* If it’s a **security breach**, isolate the affected systems.

### **Step 4: Root Cause Analysis & Resolution**

* Engineers **dig deeper** using **logs, traces, and historical data**.
* **Example tools for RCA:**
  + **ELK Stack (Elasticsearch, Logstash, Kibana)** – Log analysis.
  + **Jaeger, OpenTelemetry** – Tracing service calls.
  + **Prometheus, Grafana** – Monitoring system metrics.
* Apply a **permanent fix** and test before deploying.

### **Step 5: Post-Incident Review & Documentation**

* **Write an Incident Report** detailing:
  + What happened?
  + What was the root cause?
  + How was it resolved?
  + How can we prevent it in the future?
* Example: If a server crashed due to **high traffic**, future prevention could be **auto-scaling**.

## **Role of SLOs & SLAs in Monitoring** **SLA (Service Level Agreement)**

* A **contract** between service providers and customers specifying performance guarantees.
* Example: **99.9% uptime SLA** means a system can have only **8.76 hours of downtime per year**.

### **SLO (Service Level Objective)**

* A **targeted goal** within the SLA to ensure reliability.
* Example:
  + **SLO:** API response time should be **<200ms** for 95% of requests.
  + **SLA:** If response time **exceeds 200ms** for more than 5% of requests, penalties apply.

### **SLI (Service Level Indicator)**

* **Measured values** of SLOs.
* Example: The **actual uptime** of a service compared to the **99.9% SLA goal**.

## **Reports in Monitoring**

Reports provide insights into system health, trends, and compliance.

### **Types of Reports** Real-Time Reports

* Used for **immediate troubleshooting**.
* Example: **Live dashboards** in **Grafana, Kibana, or CloudWatch** showing real-time CPU spikes.

**Daily/Weekly Summary Reports**

* Provides an overview of system health and incidents.
* Example: **Email reports from Prometheus or Datadog** with daily uptime metrics.

**SLA Compliance Reports**

* Measures adherence to Service Level Agreements (SLAs).
* Example: Reports showing **99.9% uptime** over a month.  
    
  **Incident Reports**
* Document major incidents and resolutions.
* Example: Post-mortem analysis of a **server outage**.