Monitoring

What is Monitoring?

Monitoring is the process of collecting, analyzing, and interpreting data from various system components (applications, infrastructure, databases, and networks) to ensure they operate efficiently and reliably.

Importance of Monitoring?

- **Early Issue Detection:** Helps detect failures before they impact users.
- **Performance Optimization:** Identifies bottlenecks to improve system performance.
- **Security & Compliance:** Ensures security measures are in place and logs potential breaches.
- **Cost Efficiency:** Optimizes resource usage, reducing unnecessary expenses.
- **Better User Experience:** Ensures high availability and minimal downtime.

What Do We Monitor?

A. Application Monitoring

- **Logs:** Record events, requests, errors, and transactions.
- **Errors:** Identify application failures (e.g., HTTP 500 errors, crashes).
- **Performance Metrics:** Track request latency, response times, and throughput.
- **Tools:** Prometheus, New Relic, Datadog, AppDynamics, ELK Stack.

B. Infrastructure Monitoring

- **Servers:** Monitor CPU, memory, disk usage, and uptime.
- **Containers:** Track health, scaling, and resource consumption.
- **Cloud Resources:** Observe VM health, autoscaling, and cost management.
- **Tools:** Prometheus, Nagios, AWS CloudWatch, Datadog, Zabbix.

C. Network Monitoring

- **Latency:** Measures delay in communication.
- **Bandwidth Usage:** Monitors data transfer and congestion.
- **Security Threats:** Detects unauthorized access or DDoS attacks.
- **Tools:** Wireshark, Nagios, SolarWinds, PRTG Network Monitor.

D. Database Monitoring

- **Query Performance:** Detects slow-running queries.
- **Replication:** Monitors data consistency across instances.
- **Backups:** Ensures data recovery mechanisms are functional.
- **Tools:** MySQL Performance Schema, PostgreSQL pg_stat, Percona Monitoring.

How Do We Monitor?

A. Setting Up Alerts Based on Criticality

Incident Severity Levels (SEV1-SEV4):

- **SEV1:** Critical impact (System-wide failure, immediate attention required).
- **SEV2:** Major impact (Degraded performance affecting many users).
- **SEV3:** Minor impact (Non-critical bug, small user impact).
- **SEV4:** Informational (Warnings, minor configuration issues).

B. Defining Monitoring Strategies Across Environments

- **Development:** Debugging-level logs, flexible alerting.
- **Testing:** Performance testing, load testing.
- **Pre-Production:** Simulated production monitoring.
- **Production:** Real-time alerts, strict monitoring policies.

C. Tools Used for Monitoring

Category	Tools
Application	Prometheus, Grafana, ELK Stack, New Relic, Datadog
Infrastructure	Nagios, Zabbix, AWS CloudWatch, Splunk
Network	Wireshark, SolarWinds, PRTG Network Monitor

Database	Percona, MySQL Performance Schema, PostgreSQL pg_stat

Observability vs. Monitoring

- **Monitoring** provides insights into known issues using predefined metrics and logs.
- **Observability** focuses on debugging unknown issues using logs, metrics, and traces.
- **Key Components of Observability:**
    - **Metrics:** Numerical data points (e.g., CPU utilization, error rate).
    - **Logs:** Event records (e.g., API request logs, system logs).
    - **Traces:** End-to-end tracking of requests across distributed systems.

Incident Response & Alert Handling

Incident Response Process:

1. **Detection:** Monitoring tools identify anomalies.
2. **Alerting:** Notifies teams via Slack, Email, PagerDuty, etc.
3. **Triage:** Classify severity and assign to relevant teams.
4. **Investigation:** Debug root causes using logs, metrics, traces.
5. **Resolution:** Apply fixes and validate system stability.
6. **Postmortem:** Document learnings and improve response mechanisms.

Best Practices for Alerting:

- Avoid alert fatigue by setting priority-based alerts.
- Use AI-based anomaly detection (e.g., Datadog, New Relic).
- Implement auto-remediation scripts for common issues.

Reports & Metrics in Monitoring

A. Role of SLOs and SLAs in Monitoring

- **Service Level Agreement (SLA):** A contractual commitment between service providers and users (e.g., 99.9% uptime guarantee).
- **Service Level Objective (SLO):** Internal goal to ensure SLA compliance (e.g., API response time < 200ms).
- **Service Level Indicator (SLI):** Measurable values (e.g., Error rate = 0.1%).

B. Key Performance Metrics

Category	Metrics
Application	Latency, Error Rate, Request Rate
Infrastructure	CPU Usage, Memory Usage, Disk IO
Network	Bandwidth, Packet Loss, Latency
Database	Query Execution Time, Connection Pool Size

C. Reporting & Dashboards

- Use **Grafana, Kibana, Datadog** for real-time dashboards.
- Generate automated reports for weekly/monthly health checks.
- Correlate alerts with historical data for better insights.