

SRE-OBSERVABILITY

Q. What dashboards should we create for DevOps related information and security information?

DevOps Related Dashboards:

1. DORA metrics Dashboard
2. Build & CI/CD Pipeline Dashboard
3. Test Automation Dashboard
4. Code Quality Dashboard
5. Infrastructure Health Dashboard

Security Related Dashboards:

1. Security Incident Dashboard
2. Vulnerability Management Dashboard
3. Compliance Dashboard
4. Access & Authentication Dashboard
5. SIEM (Security Alerts) Dashboard

DEVOPS RELATED DASHBOARDS:

Q. **DORA metrics Dashboard?**

Link: [Understanding DORA Metrics. | by Rubén Alapont | Medium](#)

1. Deployment Frequency:

- ★ How quickly are we delivering new features or fixes to our customers?"(new code/updates)
- ★ often- how many times
- ★ It means how many times your team puts new code (features, bug fixes, or improvements) into the system where users can see and use it.
- ★ The more often you deploy, the faster your users get new features or bug fixes.

2. Lead Time for Changes:

- ★ time period b/w receiving order and delivering to the customer
- ★ "From the moment we start working on a feature or fix, how long does it take until it's live in production?"
- ★ it's not about the speed, it's about the quality and reliability.
- ★ A good team delivers features and fixes quickly, safely, and consistently.

3. Change Failure Rate:

- ★ It tells us how often a new code change (like a new feature or bug fix) causes problems after it's deployed.
- ★ How many times a new deployment caused a problem (like a bug, crash, or system issue) compared to the total number of deployments.

○

4. Mean Time to Recovery (MTTR):

- ★ MTTR measures how fast your team can fix a problem and bring the system back to normal after something breaks.
- ★ If your system goes down (like a website crash), and it takes your team 30 minutes to fix it and get things working again → MTTR = 30 minutes.

Q. What DORA metrics help you do?

1. Benchmark Performance: Are you faster or slower than other teams?
Compare your team with industry standards.
2. Identify Bottlenecks: Spot where delays happen in your development process.
Eg: review of code taking long/any failure in deployments?
3. Drive Continuous Improvement: Set clear goals based on numbers, and work step by step to improve them.
Eg: Reduce Lead Time for Changes from 5 days to 1 day.
4. Enhance Collaboration:

Q. How to Implement DORA Metrics?

1. Tooling: Jenkins, GitLab, CircleCI for CI/CD pipelines. These tools automatically collect important DORA metrics.
2. Data Analysis – Visualize Your Metrics: Grafana, Datadog
3. Team Involvement – Make It Part of the Culture: Regularly review DORA Metrics during: Team meetings, Retrospectives

Q. Build & CI/CD Pipeline?

1. Build Success Rate:
 - ★ Percentage of successful builds vs failed builds
 - ★ Helps to Track the stability of build process
2. Build Duration
 - ★ How long builds take on average.
 - ★ Shorter build times → Faster feedback to developers.
3. Pipeline Stage Status
 - ★ Status of stages (Build → Test → Deploy) in your pipeline.
 - ★ See where jobs are stuck or failing.

Q. Test Automation Metrics ?

1. Test Pass/Fail Rate:
 - ★ How many automated tests passed/failed during that build?
 - ★ 100 tests run → 95 passed, 5 failed → Pass rate = 95%
 - ★ High pass rate → your code working fine.
2. Test Coverage (%):
 - ★ How much of your code is tested by automated tests?
 - ★ 80% test coverage means 80% of your code has tests.
3. Flaky Tests:
 - ★ Test fails even when nothing is wrong with the code.

- ★ A test fails one time but passes next time without code change → This is flaky.

Q. Code Quality Metrics?

1. Code Churn:

- ★ Measures **how often code files are changed or rewritten** over time.
- ★ If a file is changed many times in a short period → High churn → May be unstable or need improvement.
- ★ 👍 Goal → Keep churn low for stable code.

2. Code Smell:

- ★ These are **bad coding practices** that don't break the code but make it messy or hard to maintain.
- ★ Example:
 - Duplicate code
 - Long methods
- ★ Fixing code smells makes the code cleaner and easier to work with.

3. Technical Debt:

- ★ Like “unfinished work” in code that makes future development slower or harder.
- ★ Example: Quick fixes without proper design → Adds technical debt.
- ★ Goal → Reduce technical debt over time by improving the code structure.

Q. Infrastructure Health Dashboard?

1. Server Uptime & Availability:

- ★ Measures how much time your server is up and available.

2. CPU, Memory, and Disk Usage:

- ★ Shows how much of the server's resources are being used.

3. Network Latency & Errors:

- ★ It shows how fast data moves between computers and if there are any problems (errors).
- ★ Keep the speed fast and errors very few so everything works smoothly.

SECURITY RELATED DASHBOARDS:

Q. Security Incident Dashboard?

- ★ Shows how many security incidents happened (like attacks or breaches) over time.
- ★ It helps you track problems and see how fast your team fixes them.

★ Examples are:

- Number of incidents per week/month
- Mean Time to Detect (MTTD)
- Mean Time to Respond (MTTR)

Q. Vulnerability Management Dashboard?

★ Tracks vulnerabilities (weaknesses in your system) that are found and fixed.

Eg:

- Number of vulnerabilities detected
- Severity (Critical, High, Medium, Low).
- Time taken to fix them

★ **Why It Matters:**

Helps keep your system strong by fixing security holes quickly.

Q. Compliance Dashboard?

★ Monitor if your systems follow security rules and policies.

Eg:

- Patch compliance rate (how many systems have the latest updates)
- Configuration drift (how many systems don't match the secure setup)

★ **Why It Matters:**

Helps you avoid security risks from outdated software or wrong configurations.

Q. Access & Authentication Dashboard?

★ Shows how people are logging into your system and if anything looks suspicious.

Example Info:

- ★ Number of failed login attempts
- ★ Unusual login locations or times
- ★ Privileged account usage

★ **Why It Matters:**

Helps prevent unauthorized access and detect hacking attempts.

Q. SIEM (Security Information & Event Management) Alerts Dashboard?

★ Displays security alerts triggered by your monitoring system.




★ Eg:

- Number of security alerts per day
- Most common alert types
- Top affected systems or services

★ Helps your team quickly see and respond to security threats.




Q. How to collect the metrics of **devops related dashboard** and **security related dashboard**?

1. DevOps Dashboard – Key Metrics & How We Get Them

 Metric	 What It Shows	 How We Get It
Deployment Frequency	How often code is deployed	From Jenkins, GitLab, CircleCI
Lead Time for Changes	Time from code written → code in production	From CI/CD tools
Change Failure Rate	How many deployments cause problems	From CI/CD build results
Mean Time to Recovery (MTTR)	Time to fix production problems	From monitoring tools like Grafana
Build Success Rate	% of builds passing vs failing	From Jenkins/GitLab build logs
Test Pass Rate	% of automated tests passing	From Selenium, JUnit test reports
Test Coverage (%)	% of code covered by tests	From SonarQube reports
CPU / Memory / Disk Usage	Server resource usage	From Prometheus + Grafana

Server Uptime	Time servers stay online	From monitoring tools (Datadog, Nagios)
Network Latency	Speed of data between services	From Prometheus, DataDog

2. Security Related Dashboards:

 Metric	 What It Shows	 How We Get It
Number of Security Incidents	Number of attacks or breaches	From Splunk, ELK Stack
Mean Time to Detect (MTTD)	Time to detect security problems	From SIEM (Splunk, Azure Sentinel)
Mean Time to Respond (MTTR)	Time to fix security issues	From Incident Logs (Jira, ServiceNow)
Number of Vulnerabilities Found	Number of system weaknesses	From Nessus, Qualys scanner
Severity of Vulnerabilities	Critical / High / Medium / Low	From Nessus, Qualys
Time to Remediate Vulnerabilities	Time to fix a vulnerability	From Jira, ServiceNow

Patch Compliance Rate	% of systems updated with patches	From Puppet, WSUS
Configuration Drift	Systems not following secure setup	From Ansible, Puppet reports
Failed Login Attempts	Number of wrong login attempts	From AWS CloudTrail, Auditd
Privileged Account Usage	Admin account usage	From Active Directory, CloudTrail
SIEM Alerts	Security system alerts	From Splunk, ELK Stack, Azure Sentinel