Continuous Monitoring – Part 1

This lesson provides insight into continuous monitoring.

We'll cover the following

Overview

Monitoring is implemented in different components of our distributed system to ensure the efficient functioning of these components continually over a long period of time. These components are typically the *database component, application server, web server, APIs, deployment pipelines*, and so on. Monitoring ensures the reliability and quality of the service.

Below are a few different types of monitoring systems commonly implemented in large scale distributed applications:

- Infrastructure monitoring
- Database monitoring
- Database transaction monitoring
- Data storage monitoring
- API monitoring
- Application server monitoring
- External application monitoring
- End-to-end application monitoring
- CI/CD deployment pipeline monitoring

Why monitor different components of the system?

Monitoring our system helps us understand the long-term trends by tracking the processes and events happening over a period of time. The event data that is stored to monitor a service over a period of time is known as the *time-series* data.

The analysis of this data enables a business to tweak its system for *performance, scalability, high availability*, etc. It also assists in making better *data-driven* decisions.

Monitoring when applied in the database component helps us understand the rate of growth of the size of our database. This helps in provisioning the right amount of hardware without risking the service crumbling under the heavy load.

Monitoring also helps us understand how quickly the users are growing in the system and what the latency trends of different services are. It also helps engineers track metrics like *CPU consumption, error rates, response times, disk usage, network bandwidth consumption, throughput*, and so on.

When running distributed systems, there are a plethora of tools that help us monitor our system efficiently. Typically, all the monitoring data is streamed to a web-based dashboard in the form of graphs and charts to make it easy for a decision-maker to consume.

Continuous monitoring

Continuous monitoring means all the monitoring that is implemented in the system runs in the auto mode without any sort of human intervention. If anything goes amiss, the system raises an alert and sends the notification to the concerned members of the team via email or on the *Slack* channel.

Monitoring applied in the deployment pipelines helps teams track errors that occur during the automated deployment process, such as the build fails, test failures, the build not triggering when the code is pushed to the remote repo, and so on.

Now, let's have a look at some of the popular tools that are used in the industry to monitor the distributed systems infrastructure.

Infrastructure monitoring tools

cAdvisor, Prometheus, and Grafana

These three tools, *cAdvisor, Prometheus, and Grafana* are used in conjunction to set up a monitoring system that monitors and analyzes a service.

cAdvisor is an open-source tool for running performance analysis and collecting usage data from the containers.

cAdvisor stands for *container advisor*. As its name suggests, it provides resource usage, performance characteristics, and related information about the containers running in the cloud. *cAdvisor* runs as a daemon process in the background collecting, processing, and aggregating useful *DevOps* information.

The tool has native support for *Docker*, and it enables us to track historical resource usage with histograms and so on. This helps us understand the resource consumption and memory footprint of the code running on the servers. This information helps us discover performance bottlenecks if there are any. It also helps track which processes are too hungry for memory.

When used with *Kubernetes, the cAdvisor* is integrated into the *Kubelet* binary. It is pretty intelligent to auto-discover all the containers running in the machine and collect the *CPU, memory, file system*, and *network usage* statistics. It also provides comprehensive, overall machine usage by analyzing the root container.

Let’s continue this discussion in the next lesson.