nexthink

Nexthink Architecture Quiz

Thank you very much for filling out this quiz to the best of your competences. The purpose of this quiz is to evaluate your level of expertise in architecture and in some of the technologies that we use daily @Nexthink.

Good luck and thank you for your time!

Quiz: create a Software Architecture Document (SAD)

Context description

For this exercise, we describe below a theoretical situation inspired by Nexthink with a high-level view and omitting some details for the sake of simplicity.

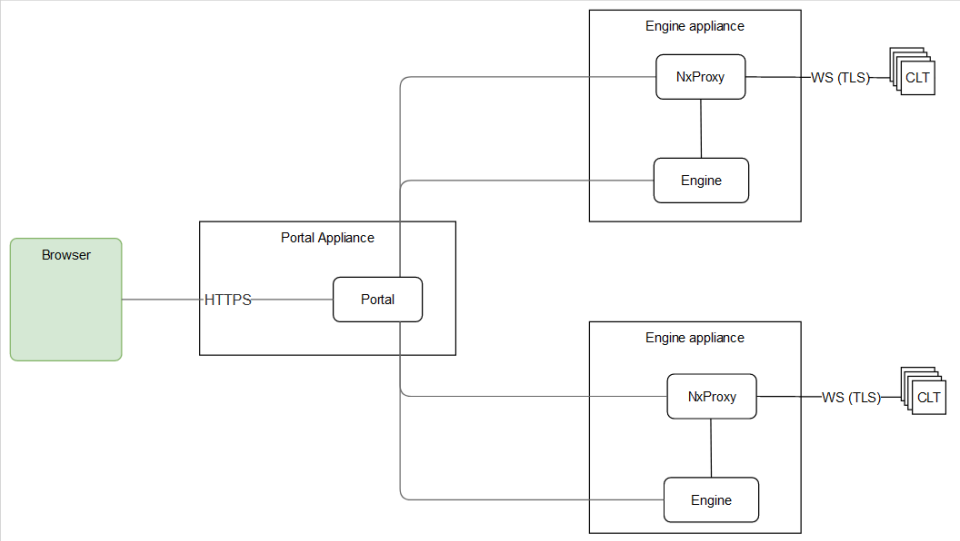
Collector

At Nexthink, we created the Collector, a lightweight software agent that is deployed on every employee device (PC or Mac) of a customer. It collects events and information such as device specifications, application versions installed and web connections, then sends them to our backend that processes the data. Each collector captures, aggregates, and broadcasts daily about 1-2000 events totaling 10MB from each device.

Our customers have between 10K and 500K installed collectors. For 500K collectors we get around 2.5TB of data per day.

For this exercise we consider a base of 1000 customers for a total of 15Million endpoints

Single-tenant back-end stack



In the single tenant back-end stack, the data is managed and kept in several in-house monolithic applications: Proxy, Engine and Portal. Deployment is made on EC2 instances in AWS, with 1 Portal appliance and multiple Engine appliances (1 each 10K collectors).

The Proxy is a home-made Java application that allows to accept and maintain communication with Collectors. It uses a web-socket to ensure bi-directional communication with the Collector. Incoming data are routed to an associated Engine (each Engine appliance do also have a Proxy).

Configuration and commands can be sent to specific collectors at the initiative of Engine or Portal (to run a command, display a survey ...)

The Engine is a home-made C++ application server with lots of business logic, which also includes an in-memory database and stores up to 200M events. This represents usually data for 2 weeks for 7k devices. You can easily see that for bigger customers, we need to scale horizontally the number of Engines.

The Portal is a home-made Java application server that offers an analytics Web front-end and computes and stores daily metrics by querying accordingly all the Engines. The Portal also offers cross-engine querying capability, the management of users and the storage of all the business content (metrics definitions, saved queries, remote actions definitions, campaigns definitions, etc...). The database for all this is a single Postgresql instance running on the Portal VM.

Target architecture requirements

The current architecture is working well but is not efficient for a Saas approach and does not support well the scale to many more development teams.
The goal is to migrate to a new architecture, fully multi-tenant, cloud-based and that allows to scale development department.

The new architecture we target for this exercise should

- Be hosted in Cloud
- Be fully multi-tenant
- Be event driven
- Allow horizontal scalability
- Allow many development teams (20+) to deliver features (more) independently with low friction
- UI should be web-based

In terms of technology stack the following elements have been retained

- AWS as the cloud provider
- Kafka for event streaming
- Clickhouse for the timeseries database
- Postgres-like DB for relational database

At product level we have the following properties

- We have interactive visualizations on aggregated data (15mn resolution)
- Access is through single page application with many parts belonging to distinct features
- We have limited number of concurrent users per tenant (10 to 200)
- We need to expose API for integrations
- Visualizations and collector actions are highly configurable by customers
- The system allows to use and correlate data of different feature in queries
- We can trigger alerts on real-time data
- For this exercise we consider a base of 1000 customers for a total of 15Million endpoints
- Data should be localized in a specific cloud region if required by the customer

Exercise description

The goal of this exercise is to propose a high-level architecture compliant with the requirements that have been exposed above.
You can propose changes to any part of the system, including protocol, flows, technologies, etc... Anything that would help in making the architecture more resilient, cost effective and match the above requirements.

Please create a Software Architecture Document (SAD) for your proposal of architecture and consider the migration path from the existing architecture to the fully multi-tenant architecture.

We do not have a strict requirement in terms of content of the SAD, please put the chapters that you think are important for us to be able to understand

- what the architecture would be and its migration path,
- the reasoning behind the architecture, the main concepts you want to achieve and the technology choices that you have done,
- what are the challenges and risks you foresee,
- how you mitigate them, and what are the assumptions that you have taken to create this architecture

Furthermore, please take into consideration constraints and potential impacts on topics like Cloud efficiency (cost), operational complexity, security and data privacy, tenant isolation and development complexity into your reasoning.

Thank you!