ORWELL

Monitorization Platform for a 5G Testbed

System Requirements and Architecture

Functional Requirements

- Collect the metrics of the machines and the network, including 5G metrics

- Mandatory use of Prometheus and Grafana (5gasp standards)

- Support for Pull and Push metrics tools

- Send reports to NetApp developers

- Send alerts to the System Admin when one of the VNFs starts to behave abnormally
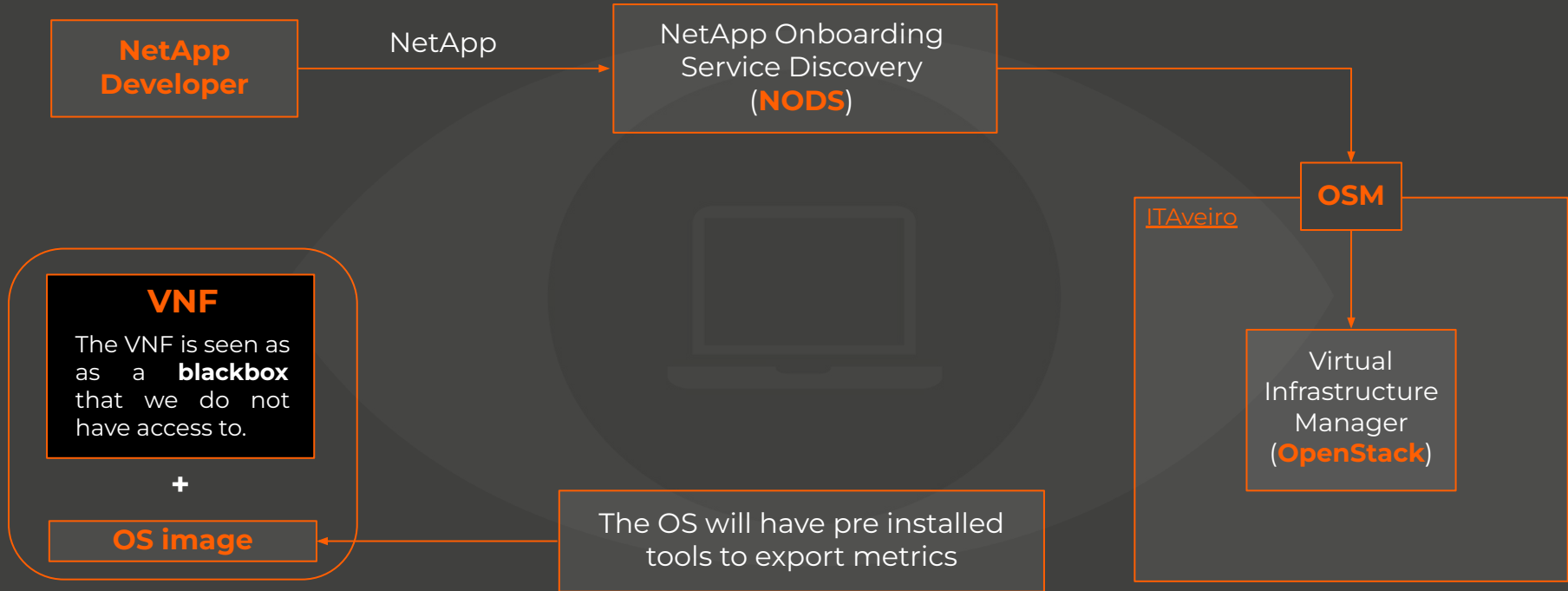
Non-functional Requirements
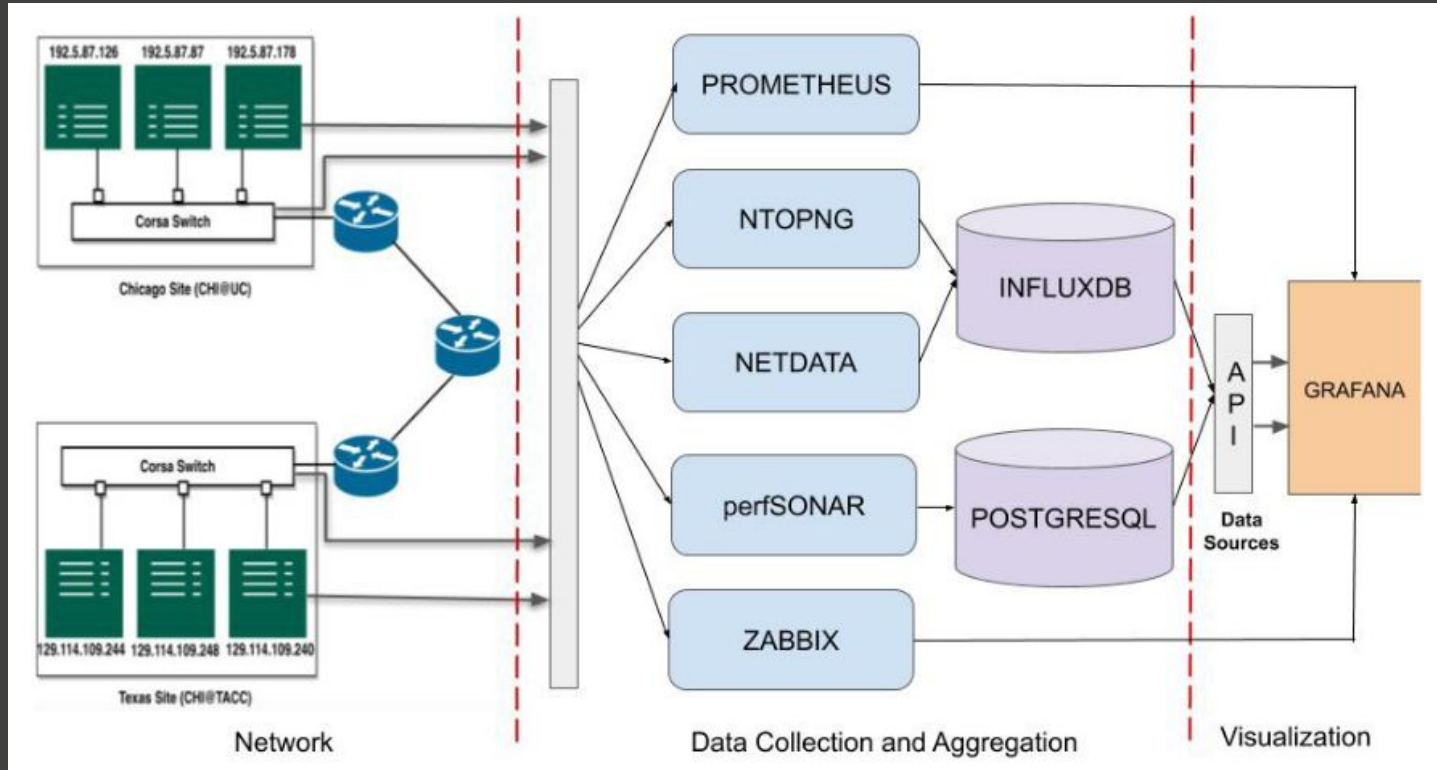
- Scalability

- Modularity

- High availability

- Easy integration of new metrics collectors

Context - 5G NetApp Deployment

NetApp Developer → NetApp → **NetApp Onboarding Service Discovery (NODS)** → **OSM**

ITAveiro

OSM → Virtual Infrastructure Manager (**OpenStack**)

VNF
The VNF is seen as as a **blackbox** that we do not have access to.

+

OS image ← The OS will have pre installed tools to export metrics
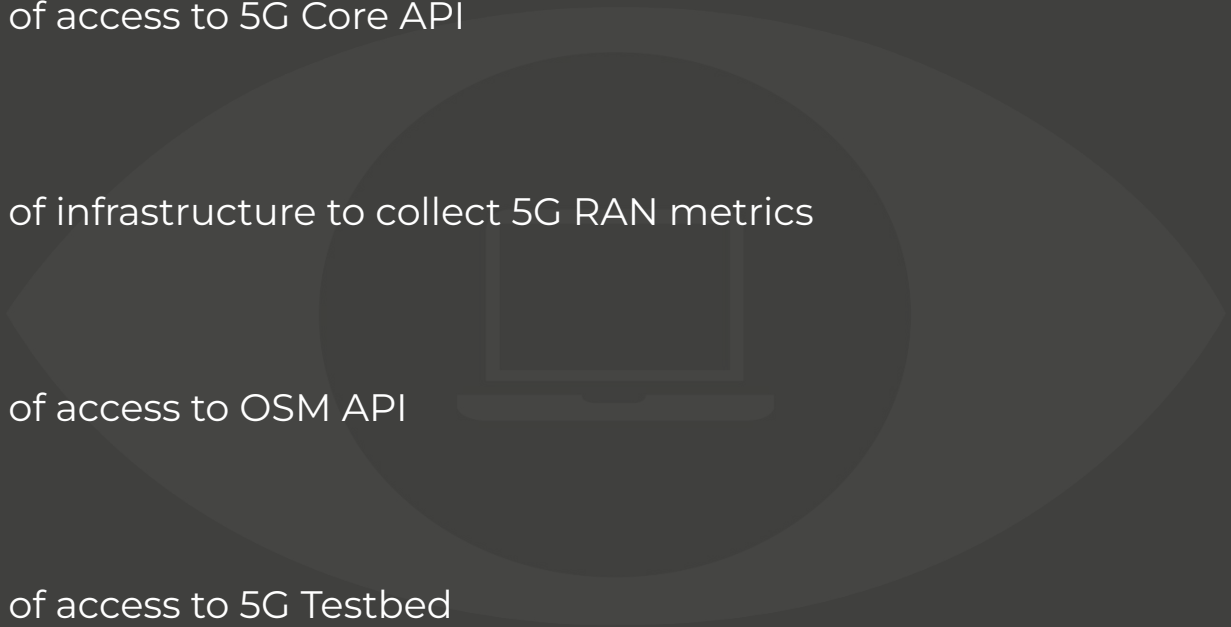
State of the Art - NetGraf

State of the Art - NetGraf Problems

_ The information is not centralized, the system has four databases (InfluxDB, PostgreSQL, Zabbix and Prometheus)

_ The collection process is invasive

_ Some metrics collectors are directly connected to Grafana

_ The API does not provide all information, so switching to another visualization system would be difficult

Risks

- Lack of access to 5G Core API

- Lack of infrastructure to collect 5G RAN metrics

- Lack of access to OSM API
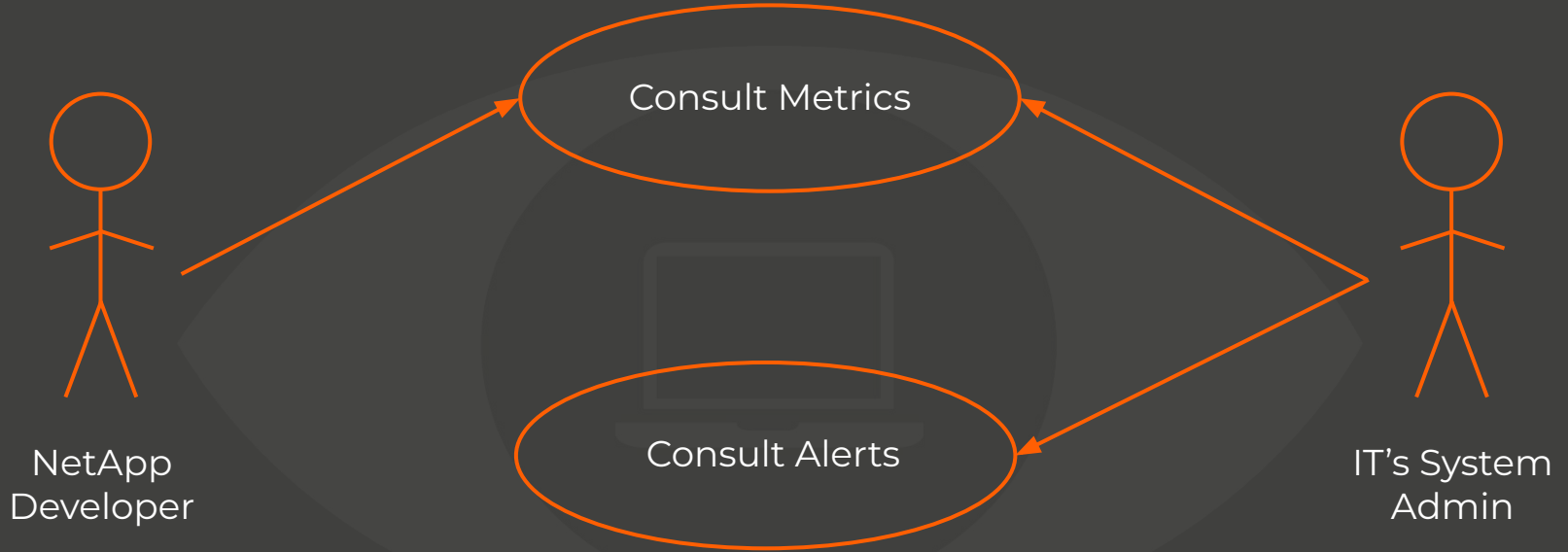
- Lack of access to 5G Testbed

Personas

NetApp Developer:

Represents the person that wants to deploy the NetApp in the testbed. He wants to know if his NetApp is successfully deployed and how it is performing.
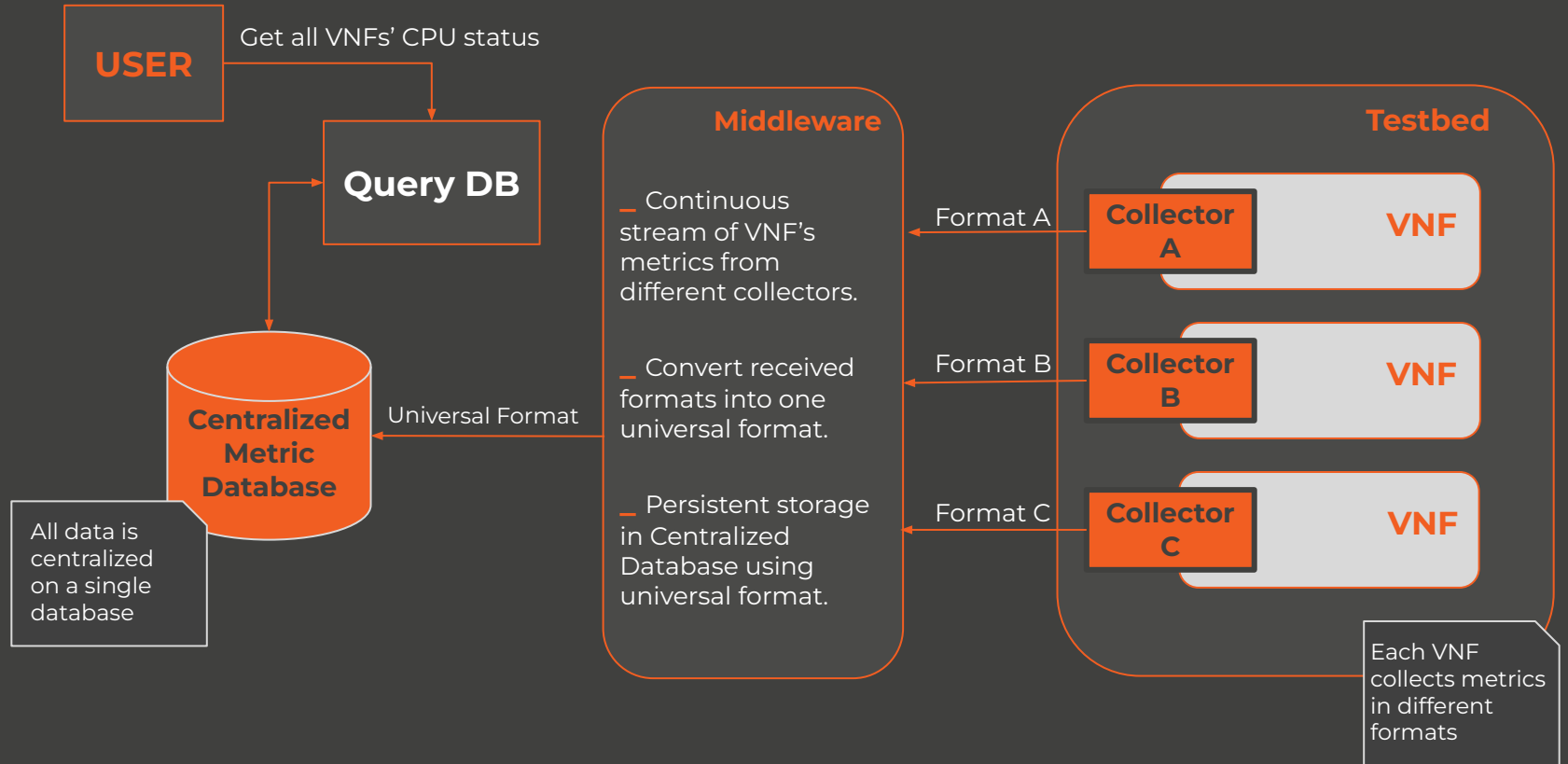
IT's System Admin:

Represents the person responsible for IT 's systems, which include the 5G testbed. He would like to know how the VNFs are performing. This can be done via our Grafana dashboard. He would also like to receive an alert when one of the VNFs starts to behave abnormally.
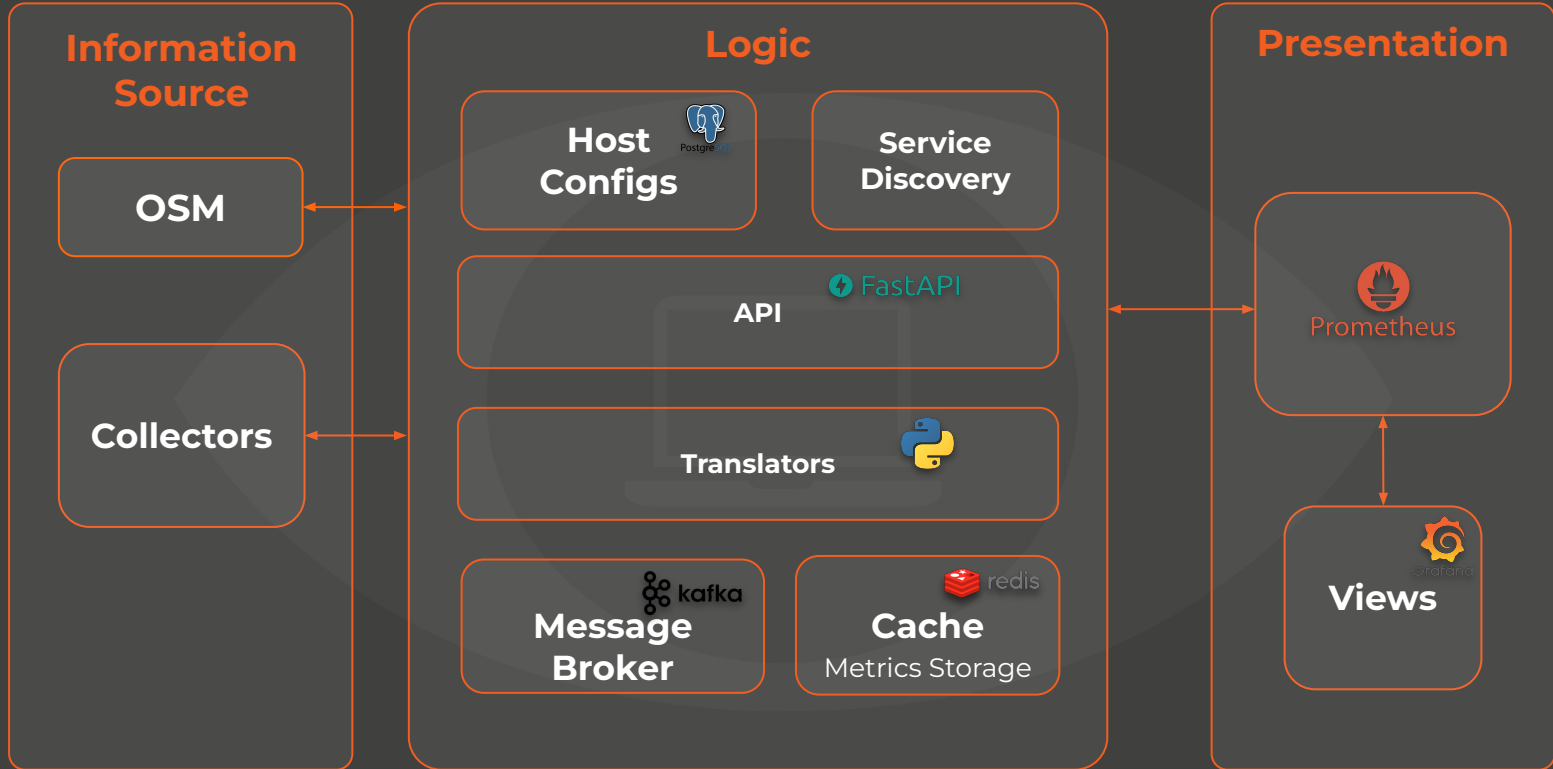
Use Cases

Our Goal

USER — Get all VNFs' CPU status

Query DB

Centralized Metric Database

All data is centralized on a single database

Universal Format

Middleware

_ Continuous stream of VNF's metrics from different collectors.

_ Convert received formats into one universal format.

_ Persistent storage in Centralized Database using universal format.

Format A

Format B

Format C

Testbed

Collector A — VNF

Collector B — VNF

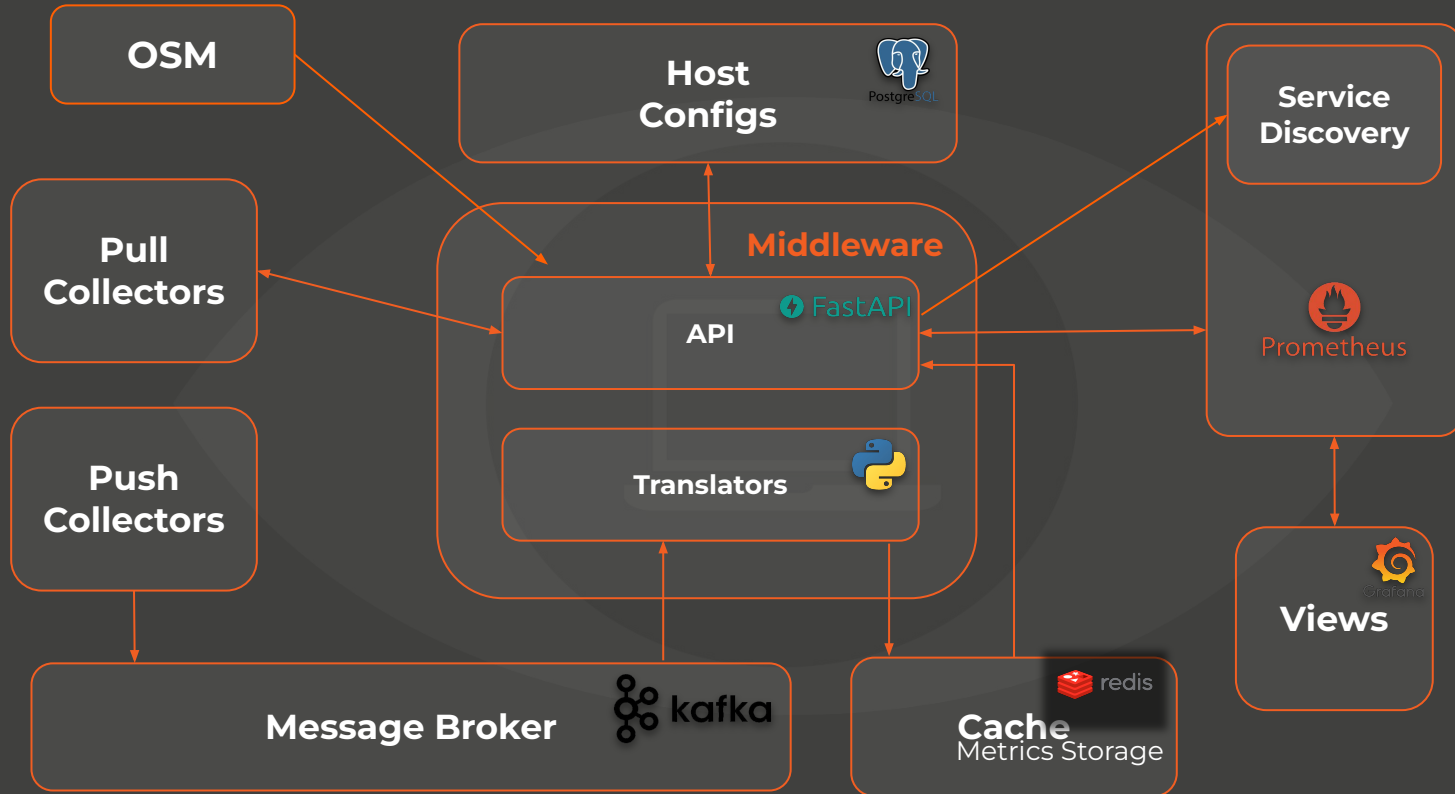**Collector C** — VNF

Each VNF collects metrics in different formats
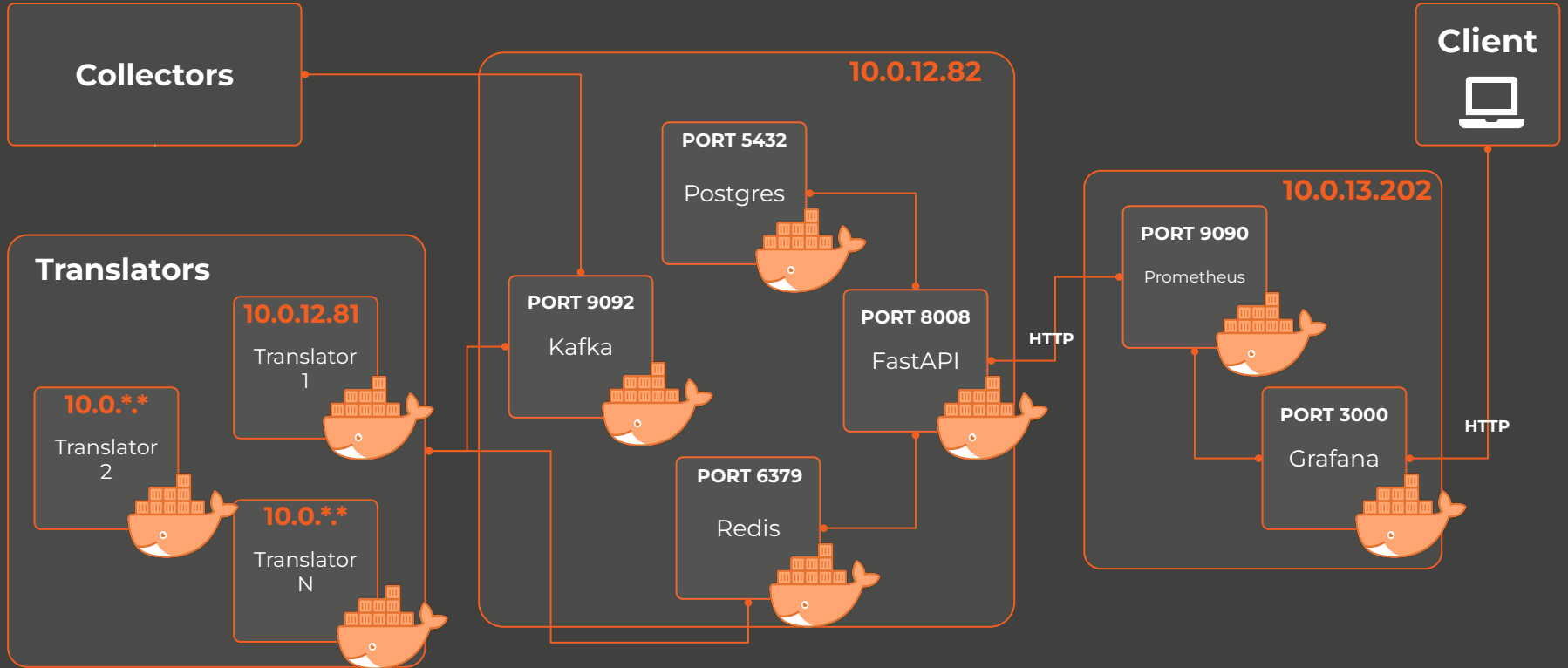
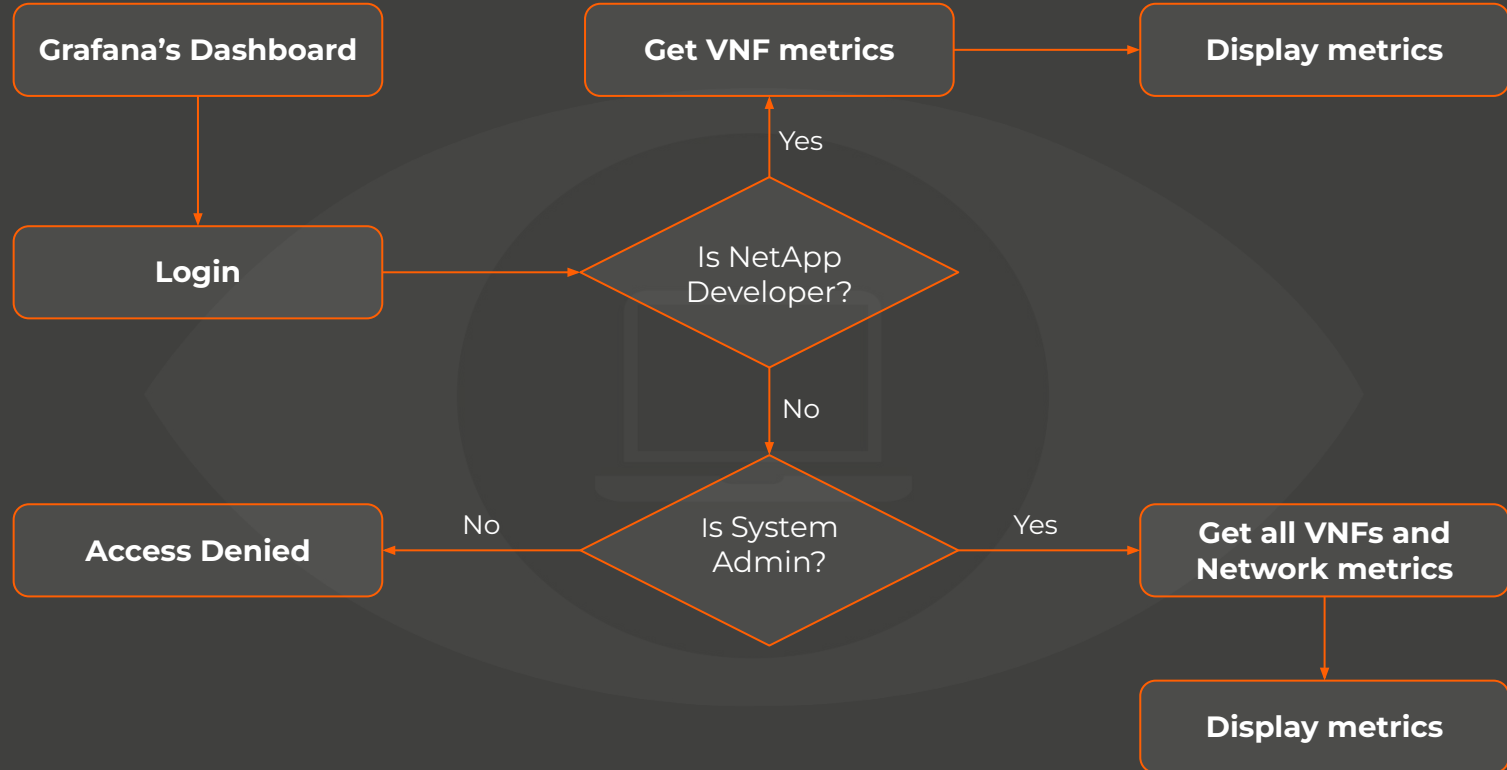# ORWELL's Architecture - Modules

ORWELL's Architecture - Interaction

ORWELL's Architecture - Advantages

_ The information is centralized and in a unique format

_ The collection process is non-invasive, we do not need to access the VNFs

_ All metrics collectors export their metrics to our middleware, which later passes them to Prometheus

_ The API provides all the information, so switching to another visualization system would be very easy

_ The system is scalable and modular

_ If the system is overloaded, more workers can be started

ORWELL's Architecture - Deployment

Collectors

Translators

10.0.12.81
Translator 1

10.0.*.*
Translator 2

10.0.*.*
Translator N

10.0.12.82

PORT 5432
Postgres

PORT 9092
Kafka

PORT 6379
Redis

PORT 8008
FastAPI

HTTP

10.0.13.202

PORT 9090
Prometheus

PORT 3000
Grafana

HTTP

Client

Flow Diagram

```
Grafana's Dashboard
        |
        v
      Login  ------->  Is NetApp Developer?
                              |  Yes
                              v
                       Get VNF metrics  ------->  Display metrics

                       Is NetApp Developer?
                              |  No
                              v
Access Denied  <--No--  Is System Admin?  --Yes-->  Get all VNFs and
                                                    Network metrics
                                                         |
                                                         v
                                                    Display metrics
```
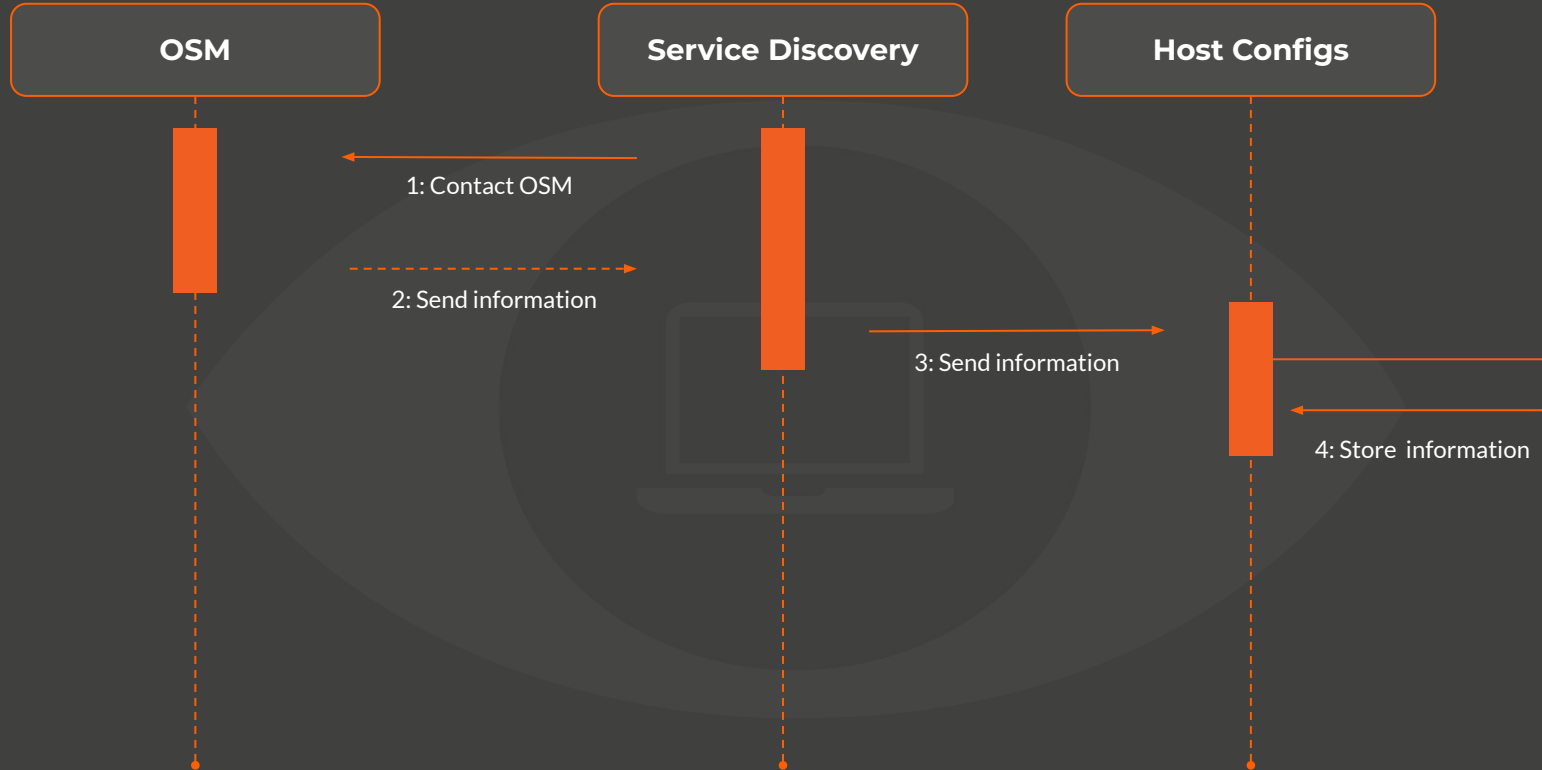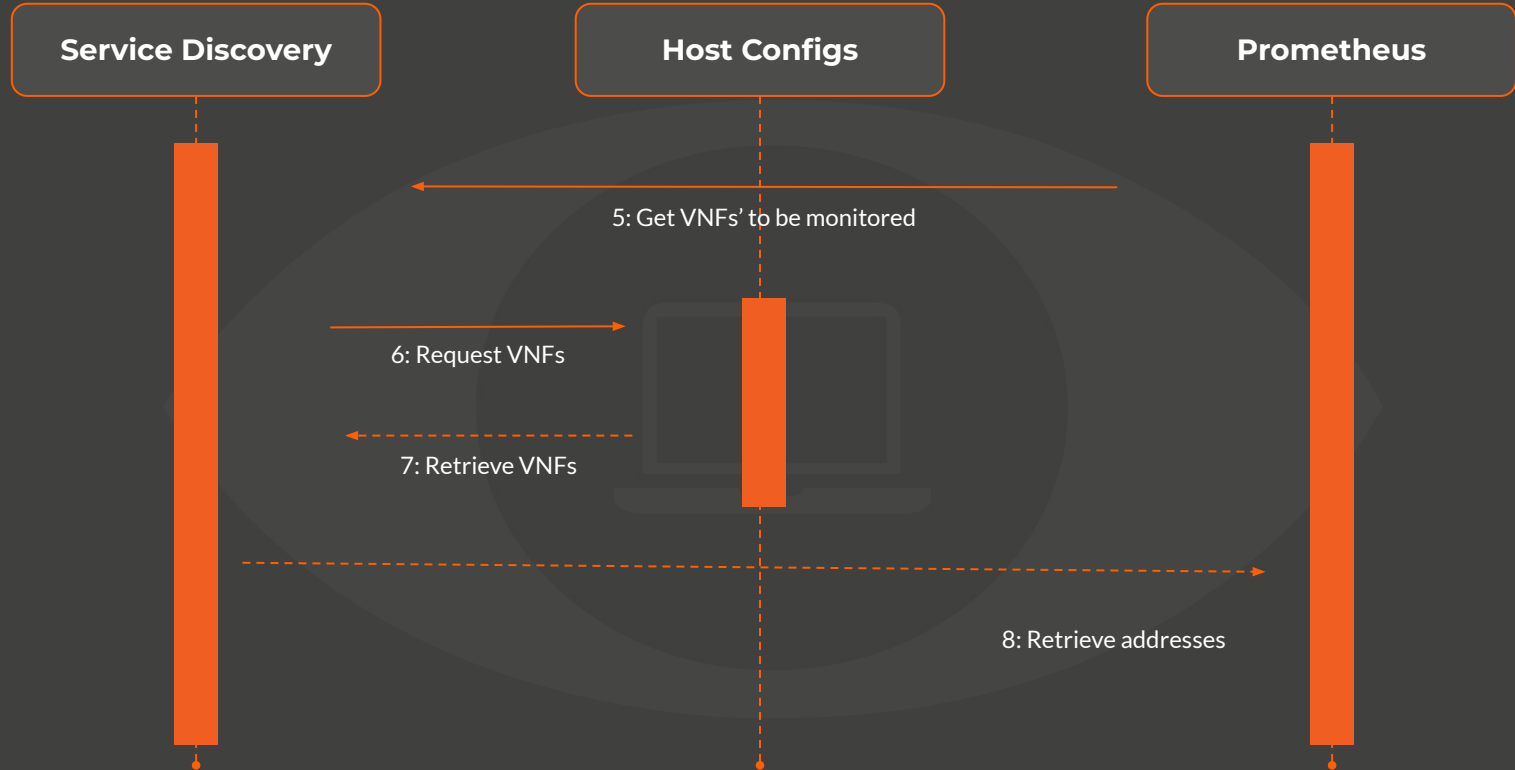
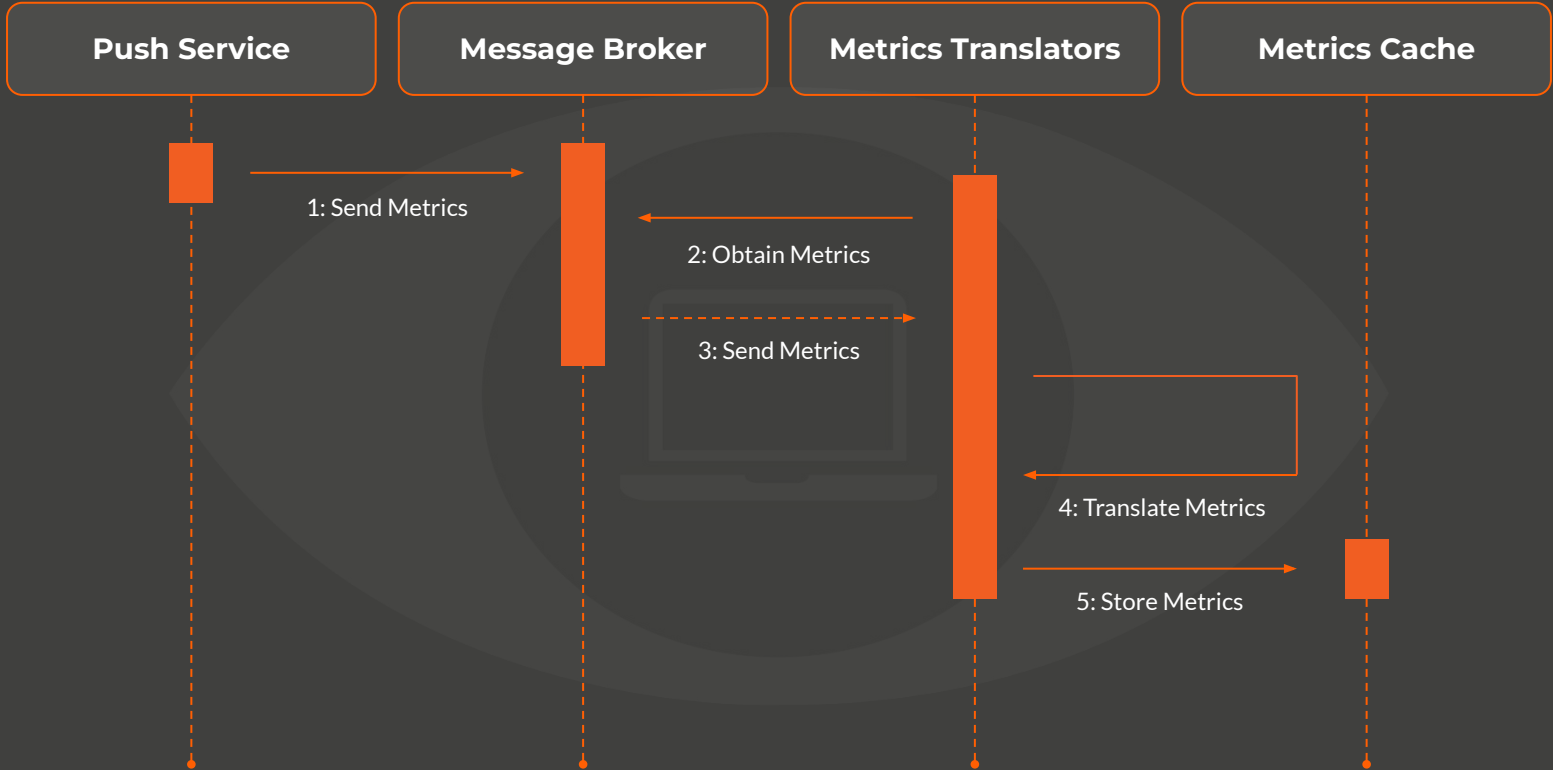Sequence Diagram - Service Discovery Initialization
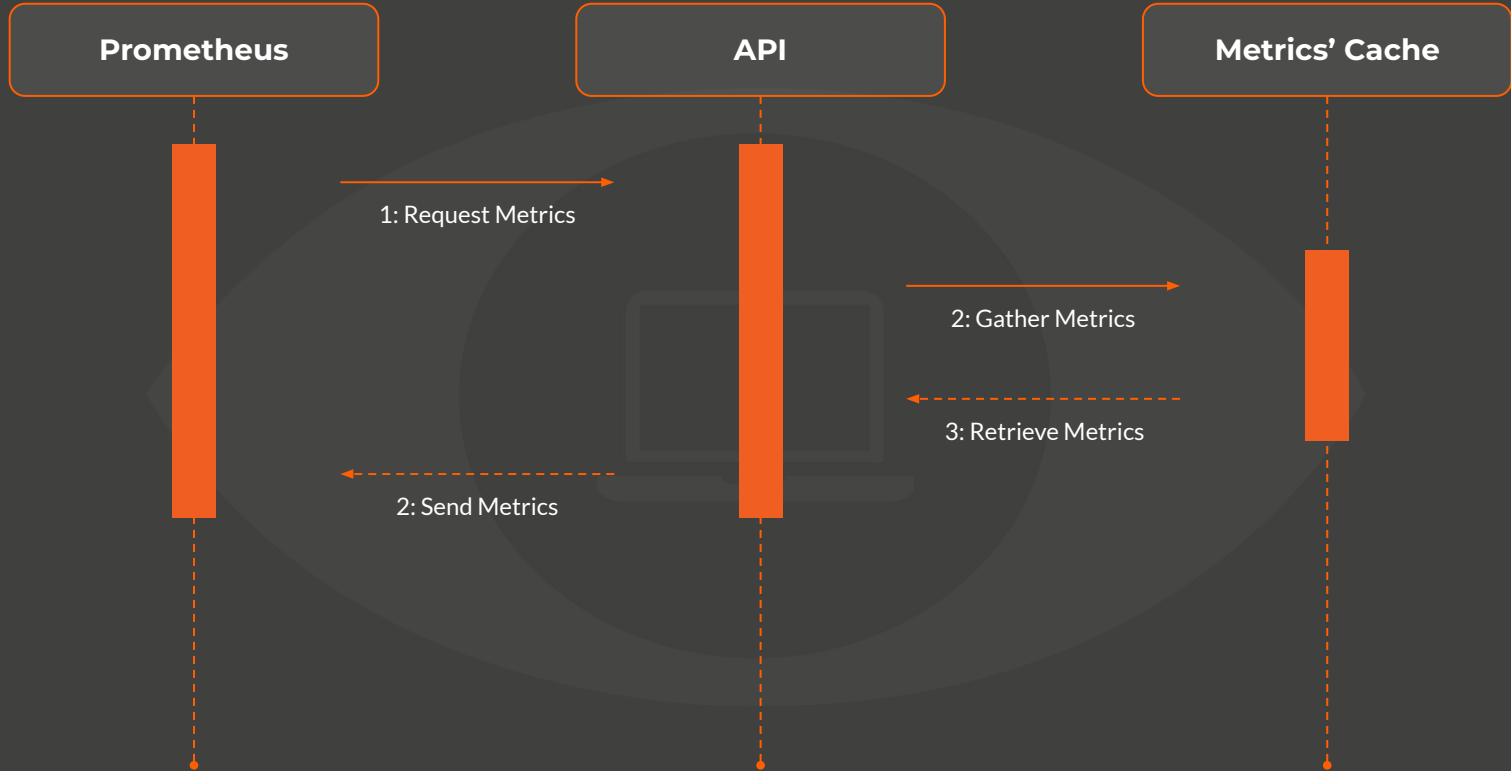
Sequence Diagram - Service Discovery Flow

Sequence Diagram - Push Metrics

Push Service   Message Broker   Metrics Translators   Metrics Cache

1: Send Metrics

2: Obtain Metrics

3: Send Metrics

4: Translate Metrics

5: Store Metrics

Sequence Diagram - Pull Metrics

Pull Service

API

Metrics Translators

Metrics Cache

1: Request Metrics

2: Send Metrics

3: Forward Metrics

4: Translate Metrics

5: Store Metrics