

Multi-Tenancy Zabbix Metrics Processing and Streaming

with Kafka, KSQL, and Grafana



OCTOBER 8 • 10, 2025
RIGA • LATVIA

About this talk

Real-world use case

The story of a solution we built and keep developing for one of our customers that had a special set of challenges and requirements.

Example of integration

Especially in larger environments, even the best of all monitoring utilities have to play the role of being just a little part of something.

Integration

A showcase of how data-driven observability can be built as built as a modern, microservice-based, extendible platform. platform.

In...Q...what?

Incubating tomorrow's IT today

Trusted Zabbix Partner & Trainer

Deep expertise from countless projects — we know Zabbix inside-out and help customers get the most out of it.

Observability for Modern IT

Modern applications have observability built in. It's the key to assuring quality, guiding development, and keeping full control of infrastructure and apps.

Automation & Cloud-Native Practices

No manual work, no fragile setups – everything must be automatable and declarative. We design microservices and integrate them with leading open-source tech.

We're a specialized IT consulting company focused IT Automation, Monitoring, Observability and Container Platforms.

We help our customers adopt and develop custom solutions, built on open-source, cloud native technologies.



Christian Anton

Owner | Consultant | Trainer



The Customer and Their Requirements

Health Care Organization

Consumes IT services from a service provider
Also runs some of their own infrastructure

Metrics View

Organization-specific metrics needed to be made available to the customer

Extendable and Multi-tenant

The solution should be capable of onboarding additional service provider's monitoring as customer-owned equipment

Shared Service Zabbix

Zabbix is used by the Service Provider to monitor all kinds of IT equipment

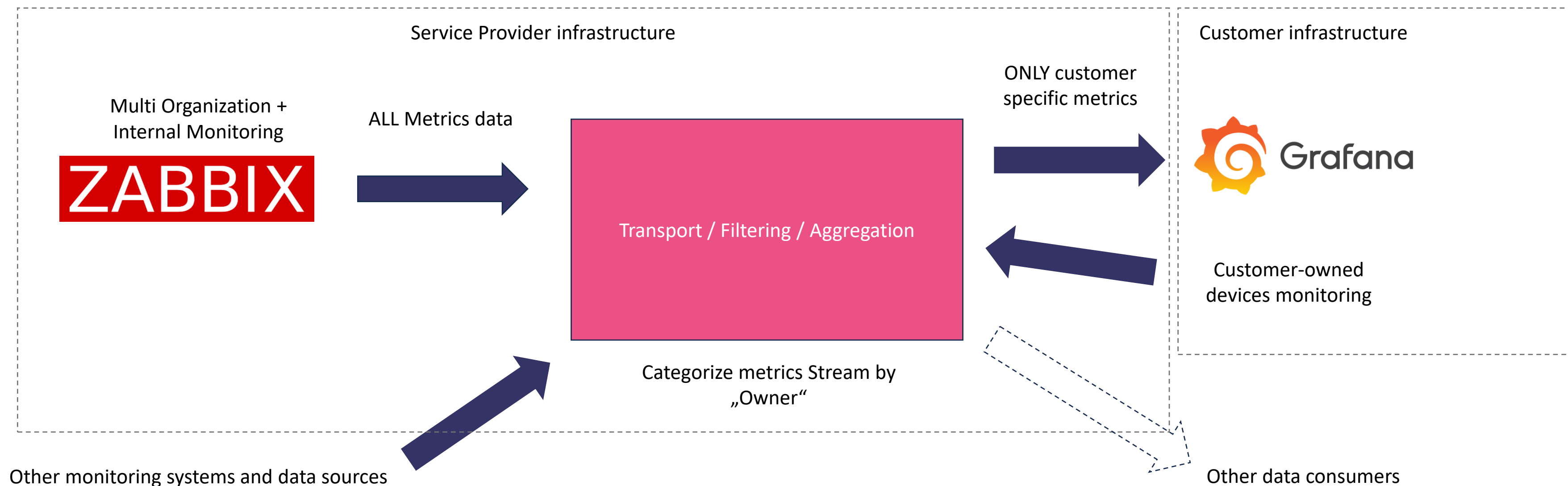
Visualization layer

Grafana was chosen as the desired platform for visualization and dashboards

Platform of choice

Stack should run in Red Hat Openshift, automated with ArgoCD

Basic Architecture



Turning the Idea into Architecture



Stream Processing

Apache Kafka – a robust and high performing data streaming platform with a large ecosystem



Target Format

Use Prometheus metrics format at the destination end - best to consume for Grafana



Kafka Deployment

Strimzi operator offers fully declarative management of Kafka Instances in Kubernetes + Openshift



Storage

Thanos is a high-performance long-term metrics store – S3 buckets downsampling, housekeeping

Getting Data into the Stack

Time-Series Data

High Volume (~6k NVPS)

Timestamp + Item-ID + Value

Datadog Vector

Future: Native HTTP Exporter + Kafka Bridge

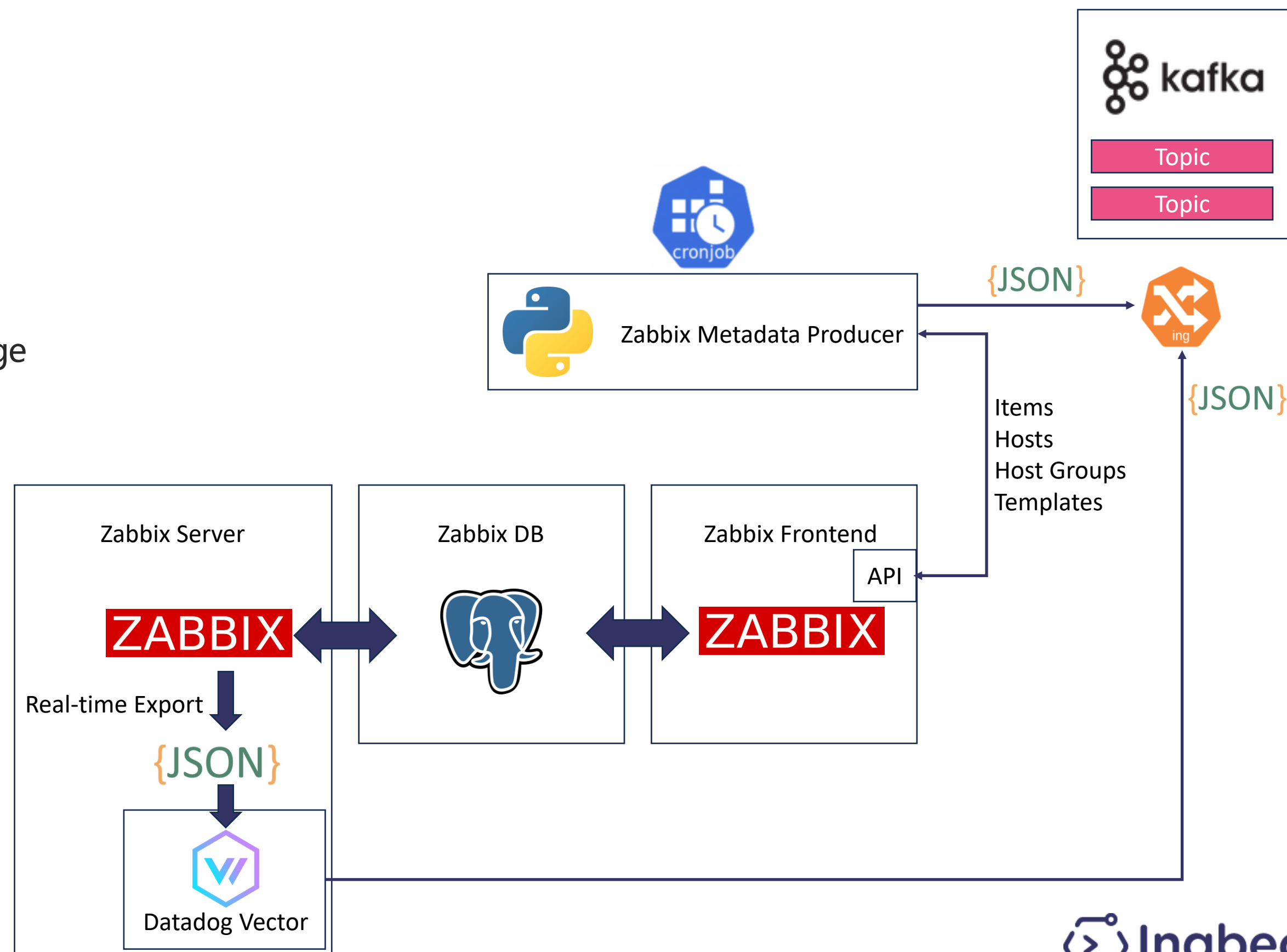
Metadata Producer

Python program, runs as Cron job

Mapping of Item-ID with Item Name, Key, Host Name, Group, Template, Tags, ...

Categorization Ruleset

- Tag selection
- Regex on Item Keys, Names
- Host Names and Host Groups



Overview
Messages
Consumers
Settings
Statistics

Seek Type

Offset

Partitions

Offset

Key Serde

All items are selected.

Value

String

16017036

+ Add Filters

	Offset	Partition	Timestamp	Key Preview
+	54214519261	2	4.10.2025, 12:54:09	16017036
-	54214407962	2	4.10.2025, 12:53:09	16017036

Key

Value

Headers

```
{
  "clock": 1759575189,
  "itemid": 16017036,
  "ts": "2025-10-04T10:53:09Z",
  "value": 10.796453
}
```

Item Value Message

Item

ItemTags 1Preprocessing 1

Parent itemsLinux by Zabbix agent

* NameCPU utilization

TypeDependent item

* Key`system.cpu.util`

Type of informationNumeric (float)

* Master itemTest Host: CPU idle time X

Select

Units%

* History

Do not store

Store up to31d

* Trends

Do not store

Store up to365d

Value mapping

Select

Populates host inventory field-None-

DescriptionCPU utilization expressed in %.

Enabled☒

Latest data

Update

Clone

Execute now

Test

Clear history and trends

```
501 - name: Test Host
502   attribute_name: _customer
503   attribute_value:
504   filters:
505   - item_attr: host_host
506     regex: testhost
507   - item_attr: hostgroups
508     regex: testgroup
509   - item_attr: key
510     regex: ^(icmp|system.cpu.util.*|agent.ping|vm.memory.utilization|vm.memory.si
```

+	103662023	2	4.10.2025, 12:35:32	16017036
+	103662240	2	4.10.2025, 12:42:28	16017036
-	103662457	2	4.10.2025, 12:45:31	16017036

KeyValueHeaders

```
{
  "itemid": "16017036",
  "hostid": "29931",
  "delay": "0",
  "key": "system.cpu.util",
  "name": "CPU utilization",
  "type": "Dependent item",
  "value_type": "numeric float",
  "value_type_num": "0",
  "master_itemid": "16017009",
  "flags": "plain",
  "history": "31d",
  "templateid": "15501385",
  "status": "enabled",
  "trends": "365d",
  "units": "%",
  "valuemapid": "0",
  "state": "normal",
  "error": "",
  "params": "",
  "description": "CPU utilization expressed in %.",
  "host_host": "testhost",
  "host_name": "Test Host",
  "hostgroups": "Linux, Linux VM, testgroup, zx - Zabbix",
  "host_templates": "Linux by Zabbix agent, Zabbix server health",
  "lastclock": "1759574709",
  "tags": "component: cpu",
  "name_resolved": "CPU utilization",
  "_customer":
}
```

Item configuration, assignment filter, metadata message

Processing Data

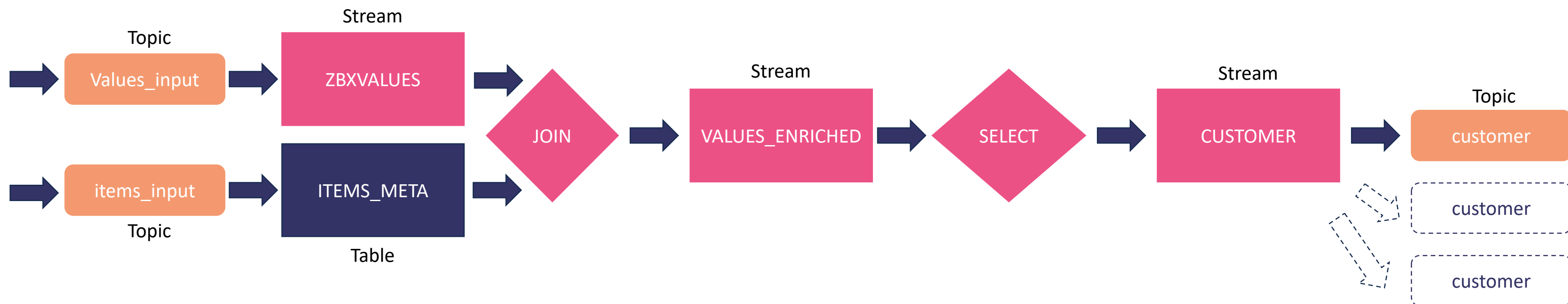
ksqlDB

SQL-like interface on top of Kafka

Generates Kafka Streams Applications internally

Real-Time Stream Processing

No JAVA Code



```
1 sqlQueries: |
2   CREATE STREAM ZBXVALUES (itemid STRING KEY, clock INT, value STRING) WITH (KAFKA_TOPIC='values_input', VALUE_FORMAT='json');
3
4   CREATE TABLE ITEMS_META (itemid STRING PRIMARY KEY, master_itemid STRING, delay VARCHAR, name VARCHAR, key VARCHAR, type VARCHAR, hostid INT, host_name
5   VARCHAR, host_host VARCHAR, hostgroups VARCHAR, host_templates VARCHAR, units VARCHAR, description VARCHAR, tags VARCHAR, _customer VARCHAR) WITH
6   (KAFKA_TOPIC='items_input', VALUE_FORMAT='json');
7
8   CREATE STREAM VALUES_ENRICHED WITH (VALUE_FORMAT='json', PARTITIONS=3) AS SELECT i.itemid AS itemid, i.master_itemid, v.clock, v.value, i.delay, i.name,
9   i.key, i.host_name, i.host_host, i.hostgroups, i.host_templates, i.units, i.tags, i.type, i._customer, 'zabbix' AS zabbix_instance FROM ZBXVALUES v LEFT
10  JOIN ITEMS_META i ON v.itemid = i.itemid;
11
12  CREATE STREAM CUSTOMER WITH (KAFKA_TOPIC='customer', VALUE_FORMAT='json', PARTITIONS=3) AS SELECT itemid AS itemid_key, AS_VALUE(itemid) AS itemid,
13  master_itemid, clock, value, delay, name, key, host_name, host_host, hostgroups, host_templates, units, tags, type, zabbix_instance FROM VALUES_ENRICHED
14  WHERE _customer = 'ourcustomer';
```

KsqlDB queries

Overview
Messages
Consumers
Settings
Statistics

Seek Type

Offset

Offset

Partitions

All items are selected.

Key Serde

String

Value Serde

String

Clear all

Cancel

16017036

+ Add Filters

Polling partitions: [0, 1, 2]
57668 ms
3 GB
30278

	Offset	Partition	Timestamp	Key Preview	Value Preview
+	917112818	1	4.10.2025, 13:23:09	16017036	{"ITEMID":"16017036","MASTER_ITEMID":"160170..."}
-	917109966	1	4.10.2025, 13:22:09	16017036	{"ITEMID":"16017036","MASTER_ITEMID":"160170..."}

Key

Value

Headers

```
{
  "ITEMID": "16017036",
  "MASTER_ITEMID": "16017009",
  "CLOCK": 1759576929,
  "VALUE": "11.081357999999994",
  "DELAY": "0",
  "NAME": "CPU utilization",
  "KEY": "system.cpu.util",
  "HOST_NAME": "Test Host",
  "HOST_HOST": "testhost",
  "HOSTGROUPS": "Linux, Linux VM, testgroup, zx - Zabbix",
  "HOST_TEMPLATES": "Linux by Zabbix agent, Zabbix server health",
  "UNITS": "%",
  "TAGS": "component: cpu",
  "TYPE": "Dependent item",
  "ZABBIX_INSTANCE": " "
}
```

Timestamp

4.10.2025, 13:22:09

Timestamp type: CREATE_TIME

Key Serde

String

Size: 8 Bytes

Value Serde

String

Size: 414 Bytes

Combined Kafka message



Getting Data Out

Metrics Receiver

Python program, running as a service

Consume data from Kafka topic

Conversion into Prometheus metrics format

Meta information as labels

Push data using Prometheus remote_write protocol

Thanos

S3 bucket (on prem) as data storage

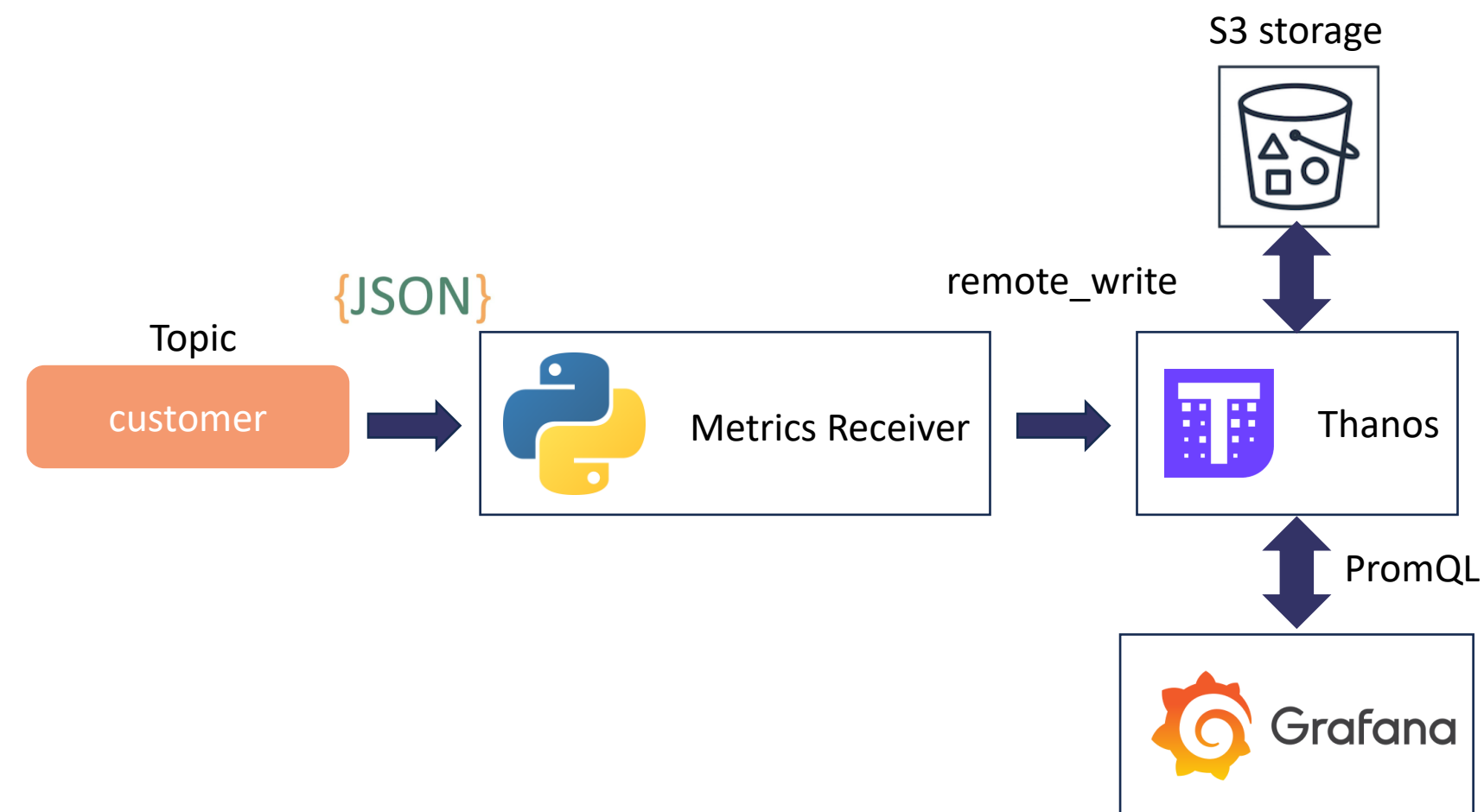
Supports downsampling, "housekeeping"

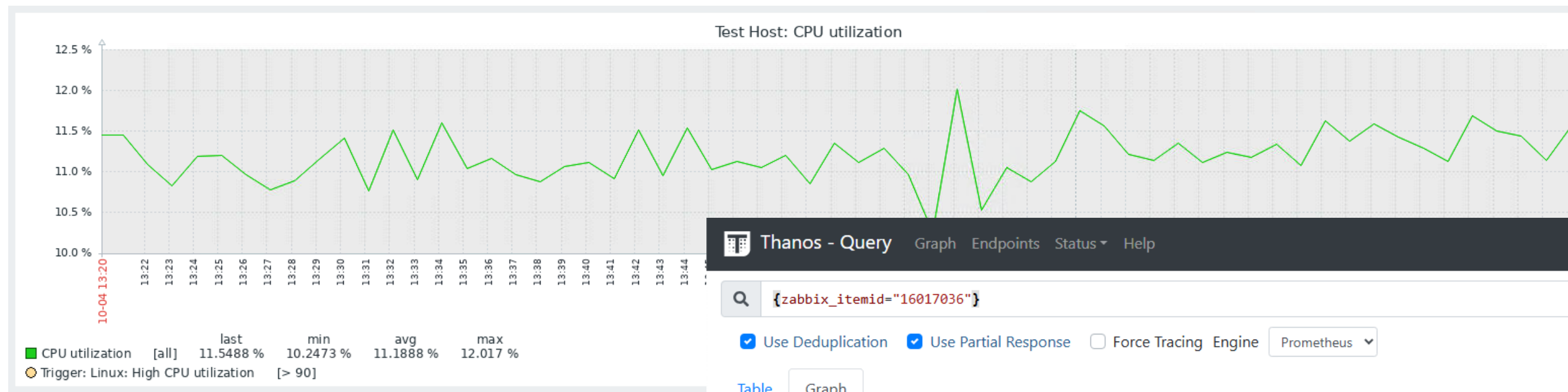
Provides Query API for Grafana

Grafana

Presentation layer

Rich dashboarding capabilities using PromQL





Thanos - Query Graph Endpoints Status ▾ Help

☐ Use local time ☐ Enable query history ☐ Enable Store Filtering

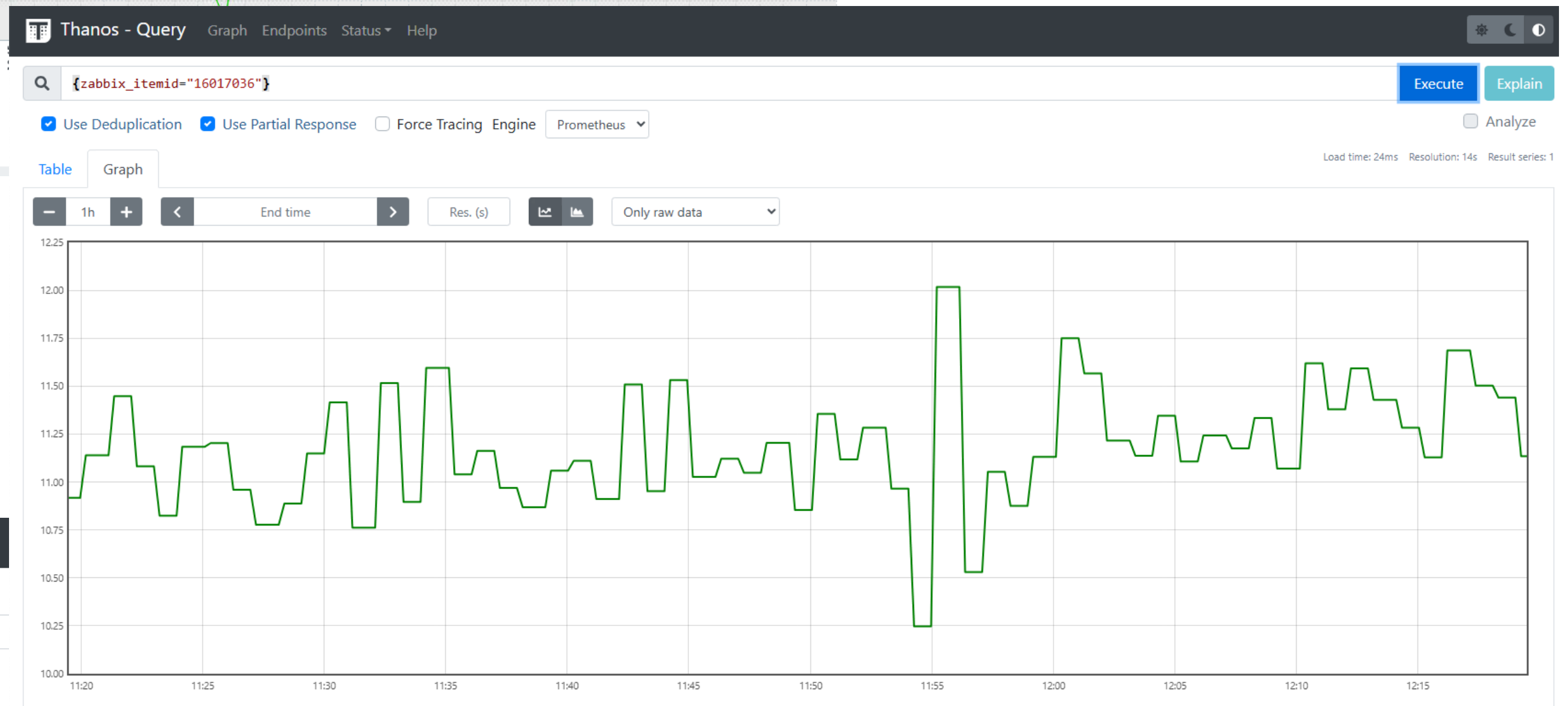
Q {zabbix_itemid="16017036"}

☒ Use Deduplication ☒ Use Partial Response ☐ Force Tracing Engine Prometheus ▾

Table Graph

< Evaluation time >

system_cpu_util{friendly_name="CPU utilization", host="testhost", host_name="Test Host", hostgroups="Linux, Linux VM, testgroup, zx - Zabbix", origin="zabbix", receive="true", tenant_id="default-tenant", unit="%", zabbix_host_templates="Linux by Zabbix agent, 10.624412000000008 Zabbix server health", zabbix_instance="sentinel770", zabbix_item_base_key="system.cpu.util", zabbix_item_key="system.cpu.util", zabbix_item_tag_component="cpu", zabbix_item_tags="component: cpu", zabbix_item_type="Dependent item", zabbix_itemid="16017036", zabbix_master_itemid="16017009", zabbix_update_interval="0"}



A single metric in Zabbix and Thanos

There's so much more...

Redis exporter

Generation of Prometheus metrics out of Redis HASH objects

Redis feeders

Query inventory data of Servers, Databases, Network Devices, Printers... from a CMDB and generate Hash entries in the Redis DB.

Also generate entries of “Locations” with Geo coordinates, addresses ...

Messaging transport bridge

A little microservice implementing a “Subscriber Model” so that users can just invite a Bot that emits specific alerts to a personal or group chat

Conclusion

Successfully

built an enterprise-ready, cloud-native observability pipeline with Zabbix as the central data generator

High-Performance

filtering and enrichment of Zabbix monitoring data to enable secure, scalable multi-tenant usage

Full Multi-Tenancy

both for additional data sources (e.g. Zabbix, Prometheus) and for customer-specific data isolation

Reusable

modular architecture that can be adapted or extended as a blueprint for other large-scale observability platforms

Thank you!!!

christian.anton@inqbeo.de