Vitrage hands-on lab

Muhamad Najjar, Eyal Bar-Ilan

CloudBand, Nokia

22-May-2018

Lab Flow

- Introduction to Vitrage
- Getting started – Vitrage CLI and UI
- Vitrage templates
- Corrective actions with Vitrage and Mistral
- Summary
- Q&A

- Link for instructions – https://pastebin.com/cem0k4cF

Lab Flow

- **Introduction to Vitrage**
- Getting started – Vitrage CLI and UI
- Vitrage templates
- Corrective actions with Vitrage and Mistral
- Summary
- Q&A

VITRAGE
an OpenStack Community Project

- Link for instructions – https://pastebin.com/cem0k4cF

Introduction to Vitrage – Root Cause Analysis Service

Project background

- Started 2.5 years ago, during the Mitaka cycle
- Became an official project 6 months later
- First official version – Newton
- ~10 active contributors in the Queens release

Vitrage Functionality

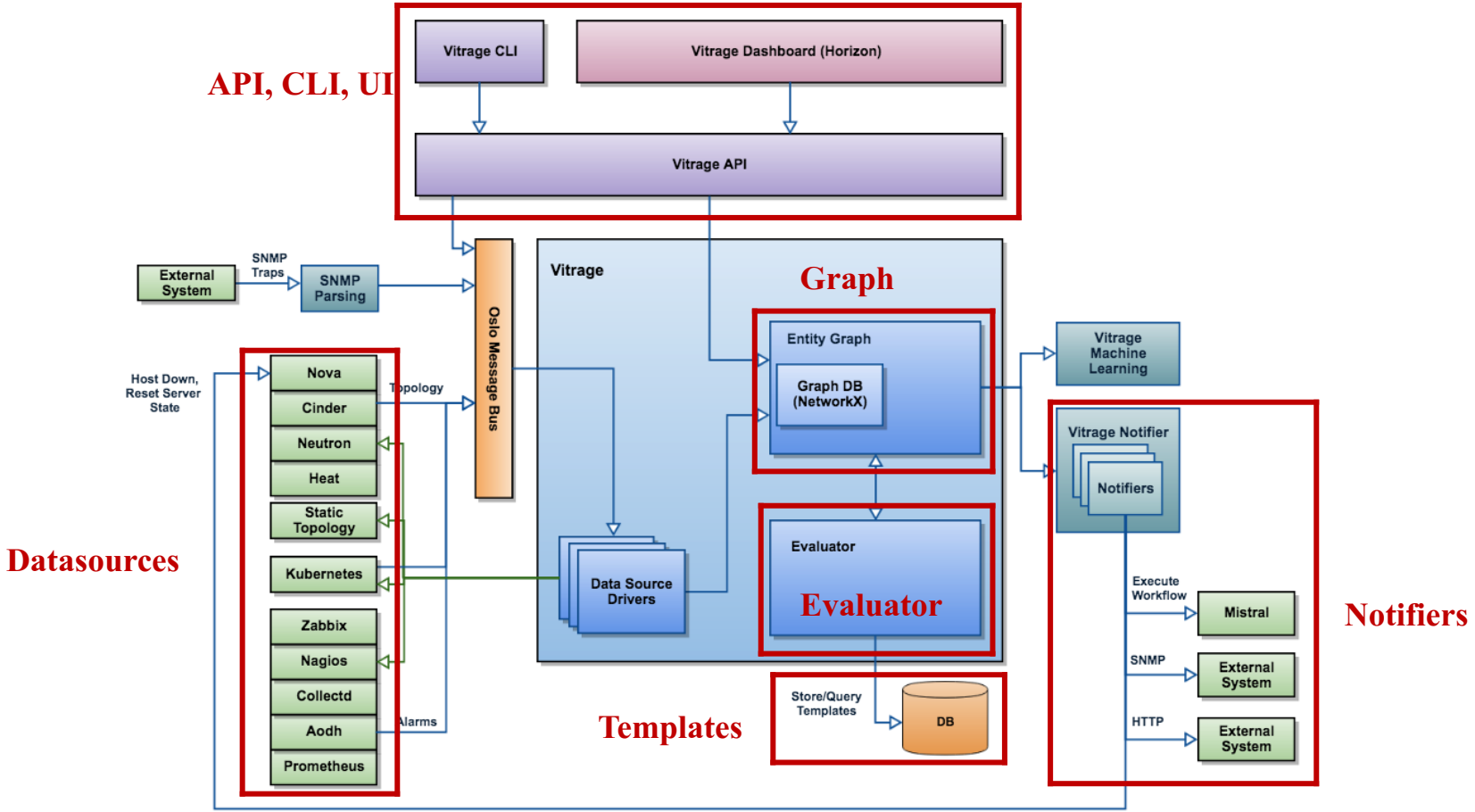
Vitrage is the **OpenStack RCA** (Root Cause Analysis) service for:

- Organizing OpenStack alarms & events
- Analyzing them
- Expanding the knowledge based on alarms & events

Vitrage's ultimate goals are:

- To give users a complete view of the structure and state of their cloud
- Allow them to address issues in a timely and effective fashion

Vitrage High Level Architecture



Lab Flow

- Introduction to Vitrage
- **Getting started – Vitrage CLI and UI**
- Vitrage templates
- Corrective actions with Vitrage and Mistral
- Summary and Q&A

- Link for instructions – https://pastebin.com/cem0k4cF

What does Vitrage UI Include?

Entity Graph

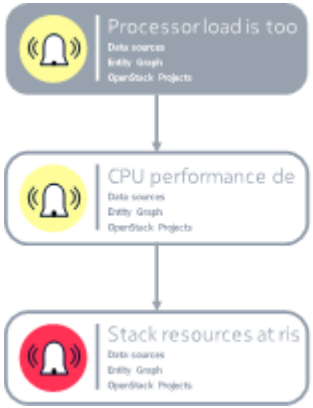
Represents the relationships between the different entities

Topology Graph

Represents system health, allowing to focuse on failing resources

Visualized RCA

Root cause analysis between alarms in the graph



Vitrage CLI

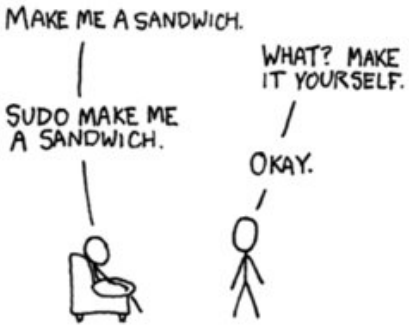
Vitrage topology show

vitrage alarm list/show/count

vitrage template list/show/add/delete

vitrage rca show

vitrage resource list/show



vitrage webhook add/delete/list/ show

Lab Flow

- Introduction to Vitrage
- Getting started – Vitrage CLI and UI
- **Vitrage templates**
- Corrective actions with Vitrage and Mistral
- Summary n' Q&A

- Link for instructions – https://pastebin.com/cem0k4cF

Vitrage Templates

- **YAML file**
  - Defines actions to be taken based on system topology & alarms
  - Added/modified by user
  - Each template can contain one or more scenarios
- **Supports the following operations** (“scenarios”)
  - if <condition> *then* raise deduced alarm
  - if <condition> *then* set deduced state
  - if <condition> *then* add causal relationship (used for RCA capability)
  - if <condition> *then* execute Mistral workflow

Vitrage Templates

- The template is written in YAML language, with the following structure:

```
metadata: ...
definitions:
    entities:
        - entity: ...
        - entity: ...
    relationships:
        - relationship: ...
        - relationship: ...
scenarios:
    scenario:
        condition: <if statement true do the action>
        actions:
            - action: ...
```

Condition building-blocks

Actions defined based on the building blocks

Vitrage Templates – Deduced Alarm Example

- If a host is in state *ERROR*, raise an alarm of type *host_in_error*

```
metadata:
   version: 2
   type: standard
   name: raise_alarm_for_host_errors
   description: host in error raises alarm
definitions:
   entities:
      - entity:
         category: RESOURCE
         type: nova.host
         state: ERROR
         template_id: host_in_error
scenarios:
   scenario:
      condition: host_in_error  # uses template_id
      actions:
         - action:
            type: raise_alarm
             target:
               target: host_in_error # uses template_ids
            properties:
              alarm_type: host_malfunctioning   # any string
              severity: critical
```

Evaluator

- Scenarios are evaluated upon system events
- Scenario conditions are represented as small graphs
- Upon event, the evaluator:
  - Retrieves scenarios relevant to event
  - Evaluates condition using sub-graph matching algorithm
  - Executes actions for each matched condition

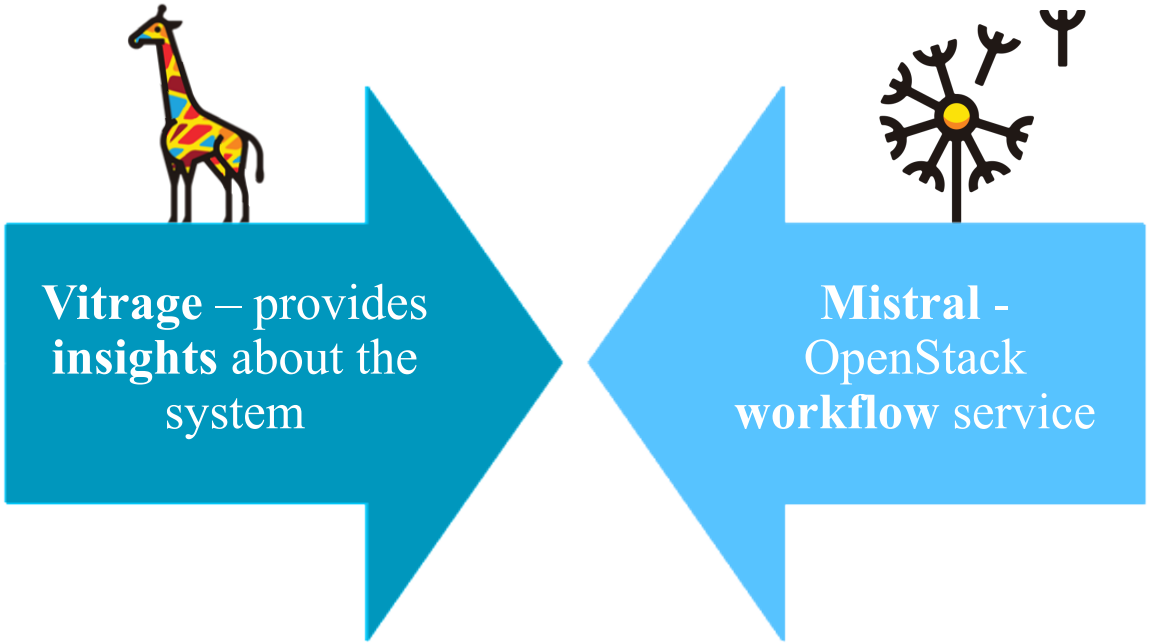
LAB

Lab Flow

- Introduction to Vitrage
- Getting started – Vitrage CLI and UI
- Vitrage templates
- **Corrective actions with Vitrage and Mistral**
- Summary n' Q&A

- Link for instructions – https://pastebin.com/cem0k4cF

Corrective actions with Vitrage and Mistral



Vitrage + Mistral: Analyze the failures **and** take corrective actions

LAB

Summary – Things We Learned Today

- How to effectively use Vitrage for fault management scenarios
- Vitrage functionality
- Vitrage architecture
- Vitrage configuration - how to adjust it to your needs
- Vitrage & Mistral for corrective actions

Questions

Come Join Us!

More Vitrage sessions in the coming days...

Vitrage wiki page: https://wiki.openstack.org/wiki/Vitrage

Official documentation: https://docs.openstack.org/vitrage/latest/

Contact info:

- Email openstack-dev@lists.openstack.org with [vitrage] tag
- IRC channel: #openstack-vitrage
- Weekly IRC meeting: Wednesday at 8:00 UTC on #openstack-meeting-4