

VAGANDO POR LAS LINDES DE LA TELEMETRIA (CON OPENNTI-NEW)

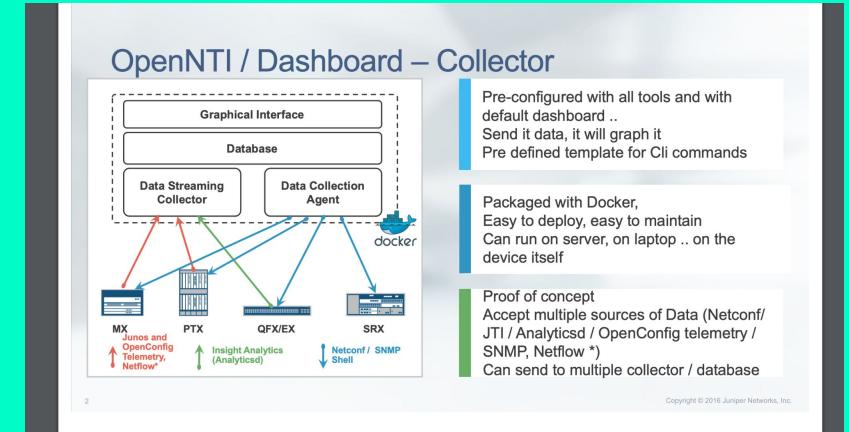
Juan P. Cerezo

¿POSIBLES USOS DE UNA
HERRAMIENTA OPEN EN UN
ENTORNO REAL?
ES RENTABLE?
ES UTIL?

ANTECEDENTES

EN EPISODIOS ANTERIORES DE ESNOG...

OPEN-NTI en el GORE-20



YouTube Premium ES

es.NOg20

Control Scaling Clipboard Displays

Search

What OpenNTI is and isn't

What it's

- An open source project
- Supported by the community
- Tool to collect and graph time series data
- Tool to demonstrate easily the value of telemetry

What it's NOT

- A Juniper "product"
- Officially supported by Juniper (NJTAC)
- A configuration management solution
- An analytics solution
- A replacement for CAE

MEDIALAB PRADO

10:02 / 57:59

ESNOG20_OPEN-NTI

This screenshot shows a YouTube video player. The video title is 'es.NOg20'. The main content area displays a slide with the heading 'What OpenNTI is and isn't'. It is divided into two sections: 'What it's' and 'What it's NOT'. The 'What it's' section lists four bullet points: 'An open source project', 'Supported by the community', 'Tool to collect and graph time series data', and 'Tool to demonstrate easily the value of telemetry'. The 'What it's NOT' section lists five bullet points: 'A Juniper "product"', 'Officially supported by Juniper (NJTAC)', 'A configuration management solution', 'An analytics solution', and 'A replacement for CAE'. On the left side of the video player, there is a thumbnail image of a man speaking at a podium. The podium has a sign that reads 'MEDIALAB PRADO'. The video player interface includes standard controls like play/pause, volume, and a progress bar indicating 10:02 of 57:59. The overall background of the slide is white with blue headers and callouts.

QUE HA CAMBIADO?

- La idea original se mantiene
 - actualización de versiones
 - nuevas funcionalidades
 - mismo soporte
 - mejoras en el despliegue y
usabilidad?
-

DECONSTRUCCIÓN

open-nti-new

- proyecto personal
 - modularizar los componentes
 - optimizar la selección de escenarios
 - mismo soporte (best effort)
 - mejoras en el despliegue y usabilidad (SI!)
-

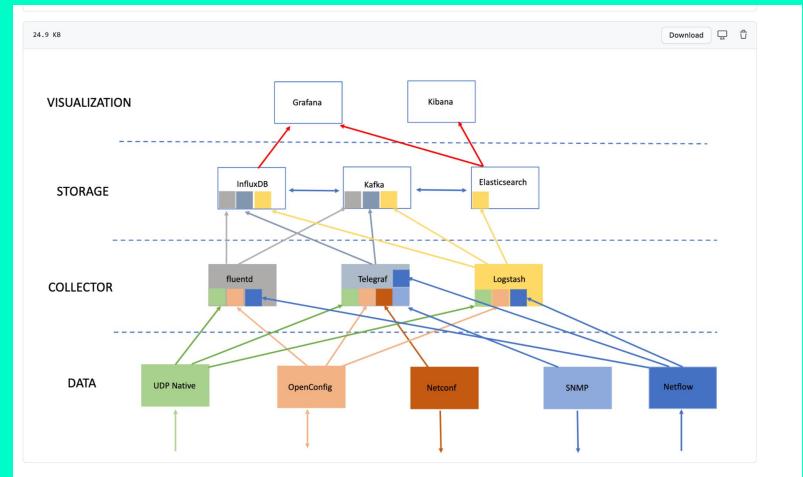
DECONSTRUCCIÓN

open-nti-new

The screenshot shows a GitHub repository page for 'open-nti-new'. At the top, there's a search bar and navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the header, there are tabs for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', and 'Insights'. The 'Code' tab is selected. It displays a list of commits from 'Pablo' on 'master'. The commits are:

- adding docker-compose-logstash file (first commit, 2 years ago)
- docs (first commit, 2 years ago)
- inputs (first commit, 2 years ago)
- outputs/influxdb (first commit, 2 years ago)
- web_ui/grafana (first commit, 2 years ago)
- Makefile (first commit, 2 years ago)
- docker-compose-fluentd.yml (first commit, 2 years ago)
- docker-compose-logstash.yml (adding docker-compose-logstash file, 2 years ago)
- docker-compose-telemetry.yml (first commit, 2 years ago)

On the right side, there are sections for 'About', 'Releases', 'Packages', and 'Languages'. The 'About' section says 'openNTI deconstructed' with 2 commits. The 'Languages' section shows Python (65.0%), Ruby (20.0%), Makefile (7.0%), Dockerfile (4.0%), and Shell (2.5%).



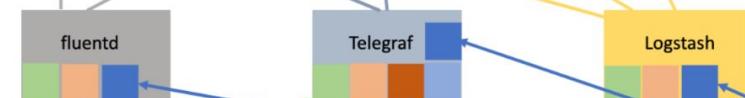
VISUALIZATION



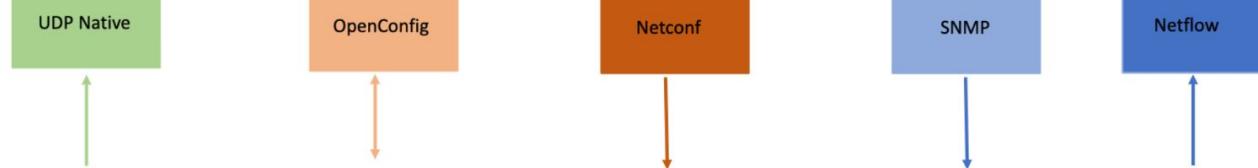
STORAGE



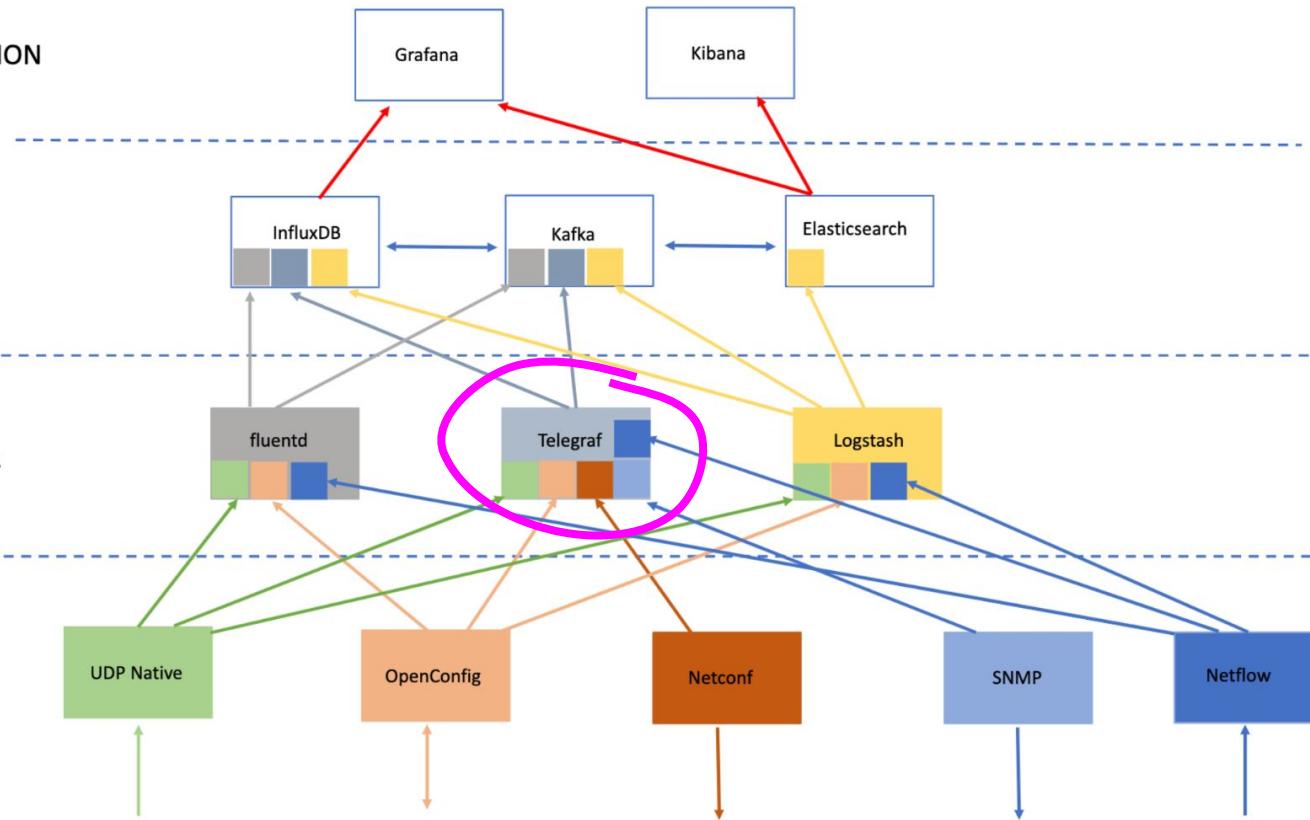
COLLECTOR



DATA



VISUALIZATION

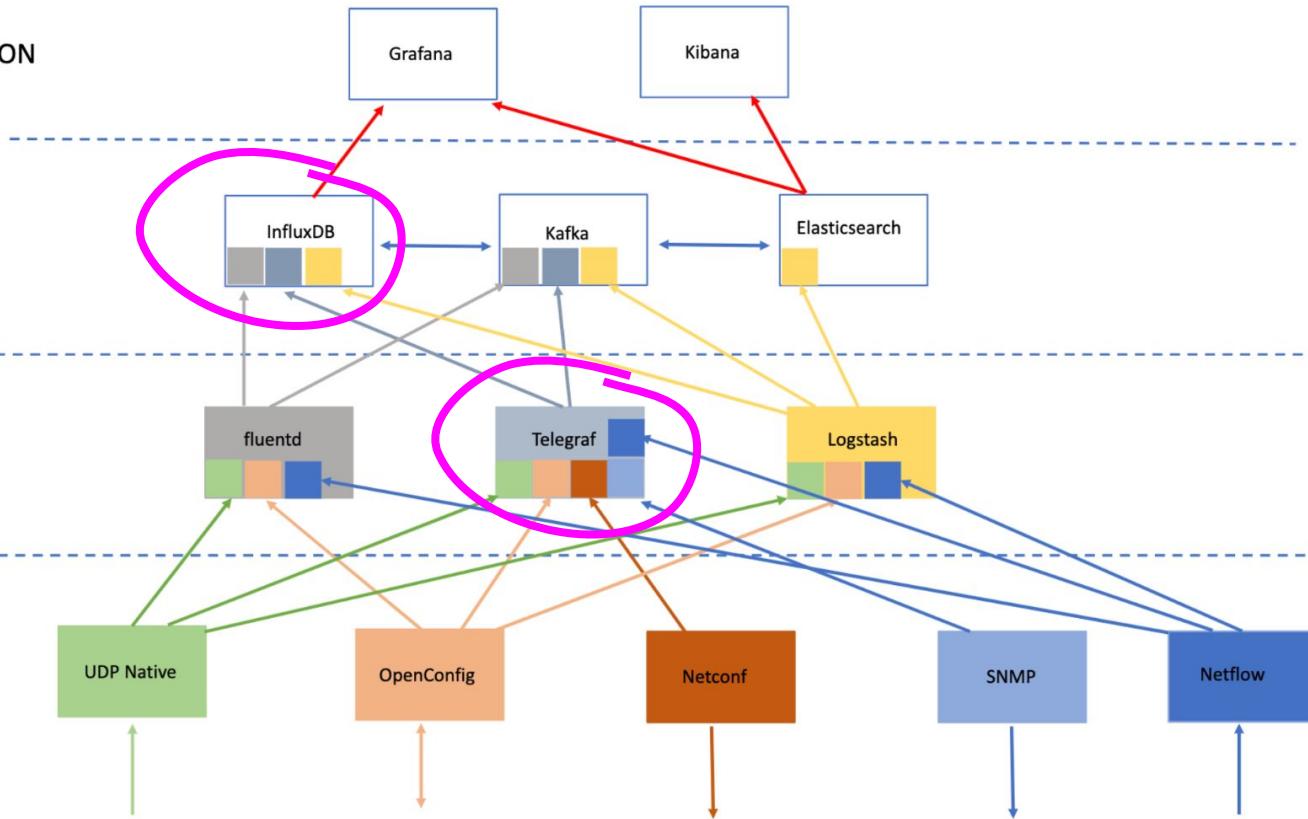


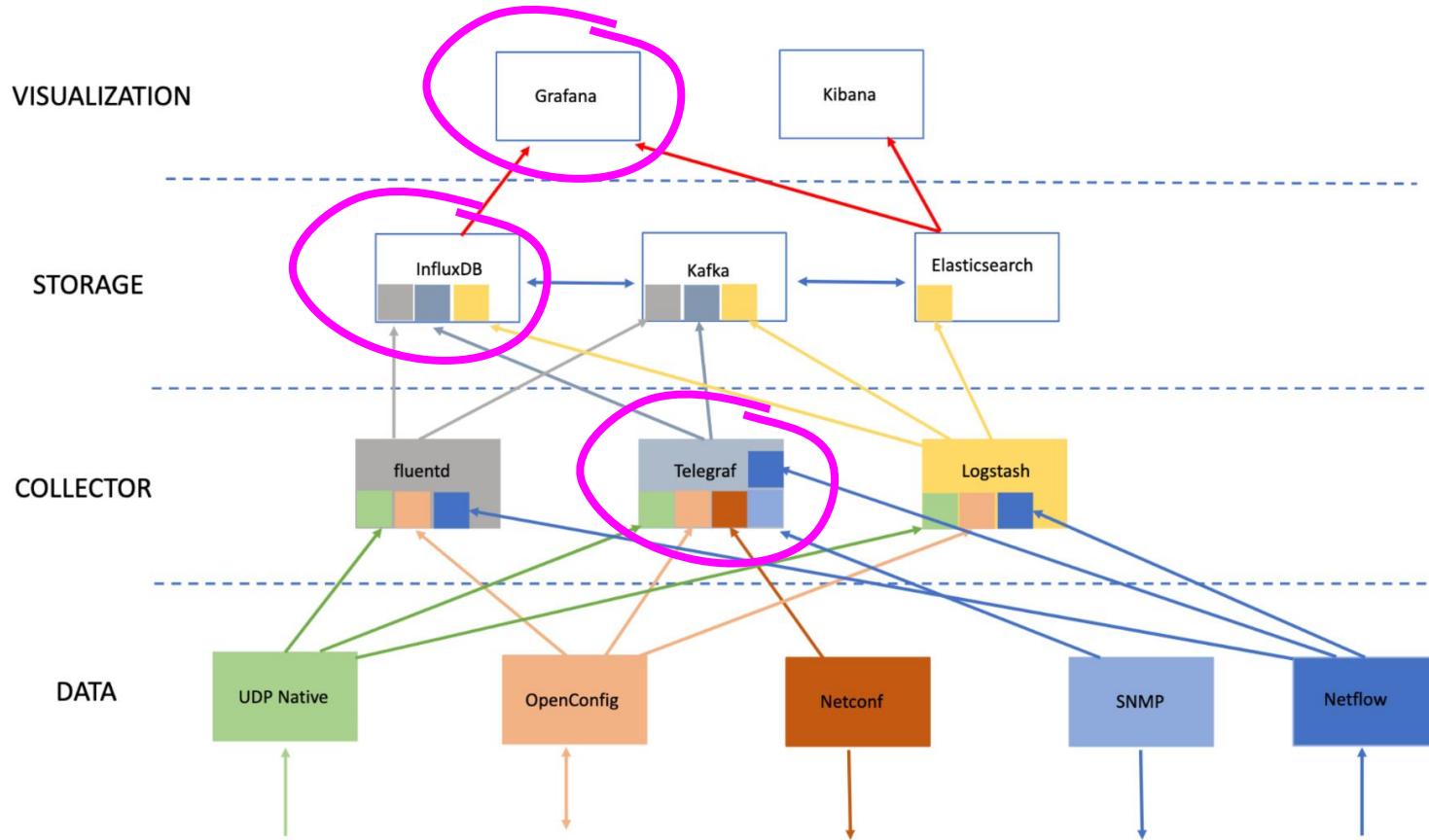
VISUALIZATION

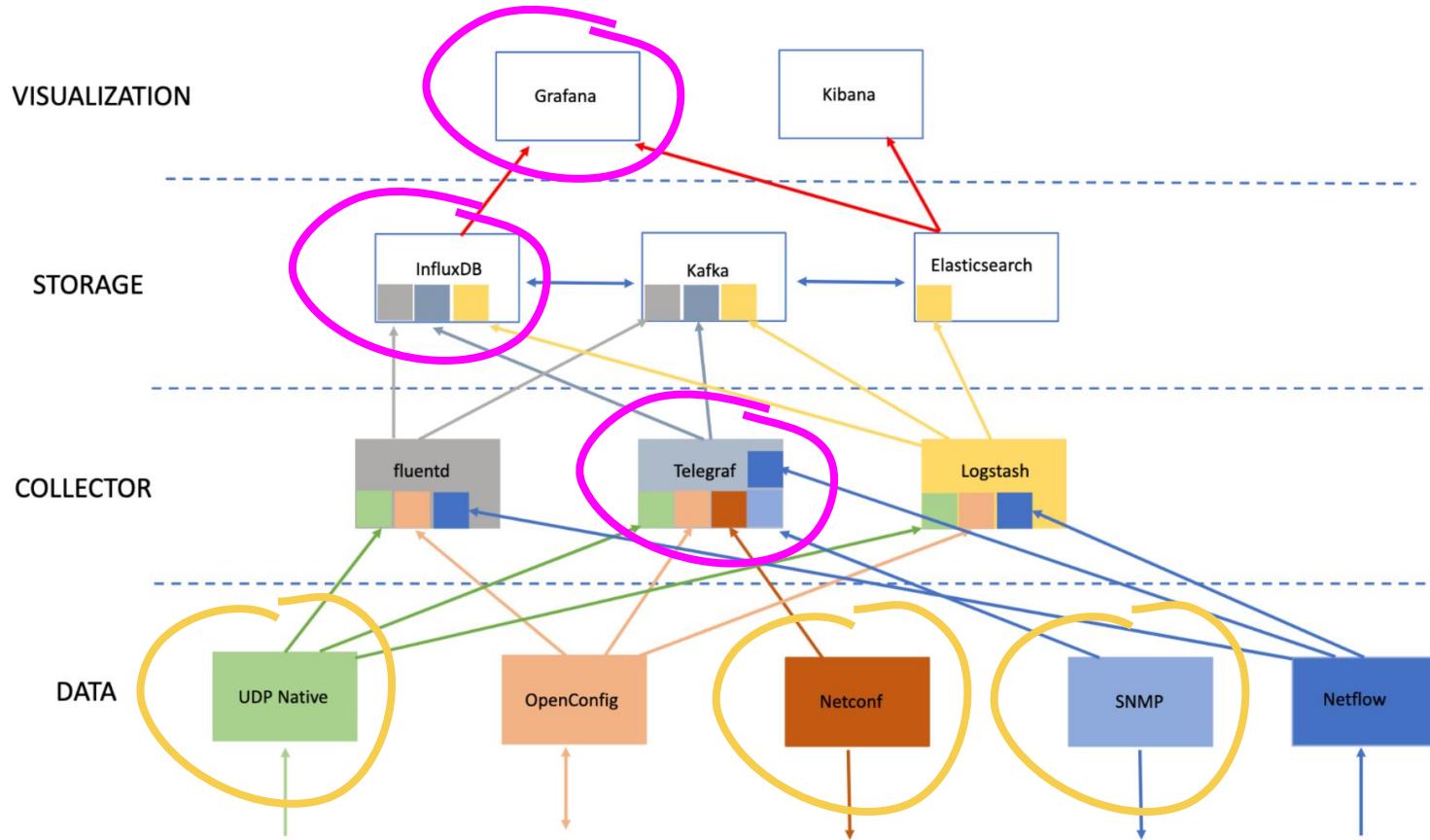
STORAGE

COLLECTOR

DATA







PUESTA
EN
MARCHA

MATERIAL

Lo tienes todo en casa

- una red de equipos (de red)
- docker
- git, python
- code editor & test
- un equipo sobre que trabajar
(preferiblemente ubuntu)

HIPÓTESIS

REQUERIMIENTOS

1. Red ISP de unos 150 equipos (Juniper) de red
2. Login (2 diferentes)
3. Diferentes funcionalidades a monitorizar: **BNG, CG-NAT, IXP, chassis & RE**
4. Posibilidad de realizar un chequeo de red permanente (**HealthCheck**) durante periodos específicos
5. Ortogonal a otras mediciones o servicios de monitorización



INSTALACION

Y

PUESTA EN MARCHA

ADAPTACIONES

Personalización de la base de equipos

Usando los formatos habituales, configurando hosts, grupos, autenticación... para el tipo de colector/ingestión de datos seleccionado

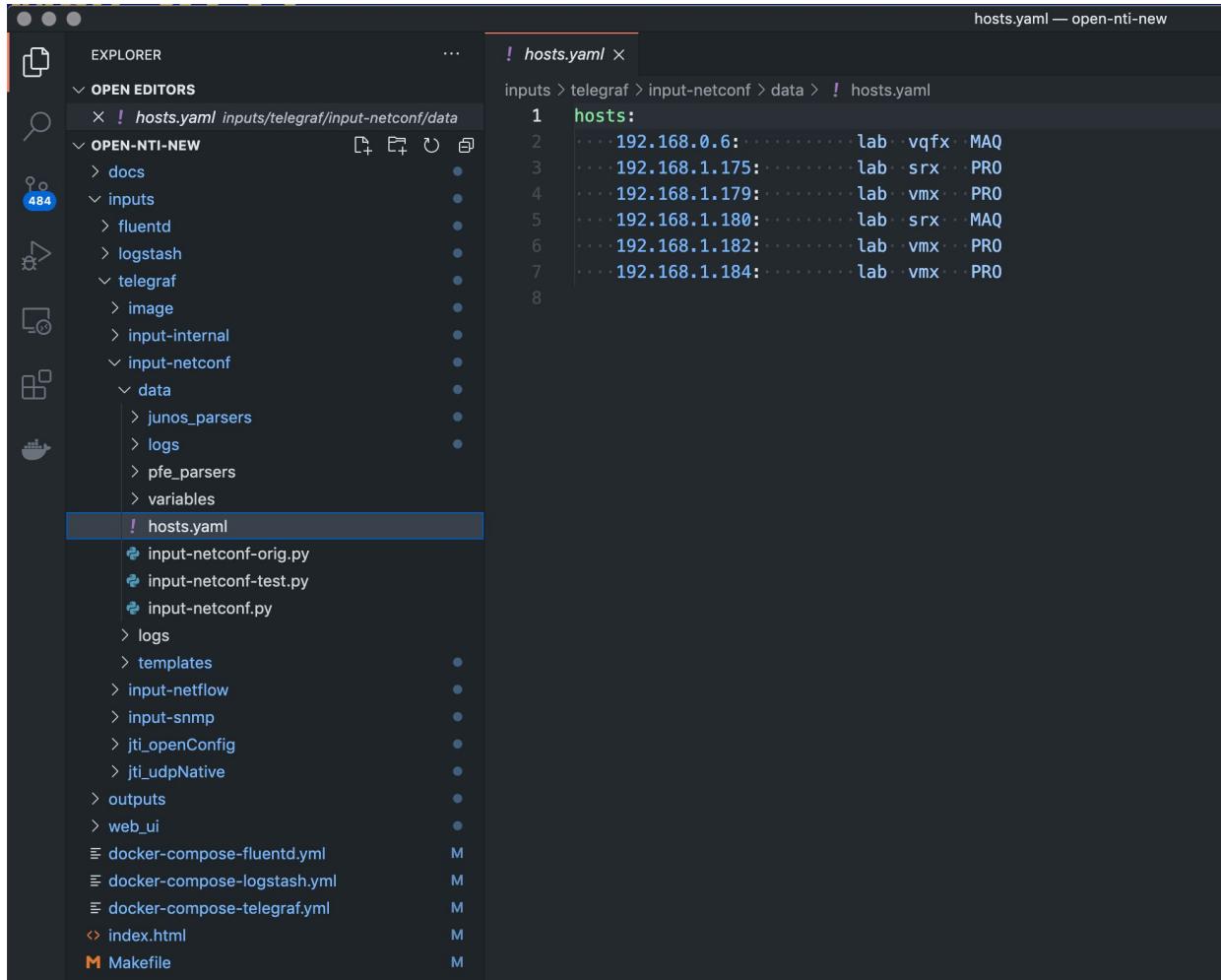
Personalización de los KPIs

- Parámetros genéricos de red a monitorizar
- Parámetros específicos para cada grupo de funcionalidades: IXP/routing, BNG, CG-NAT, babysitting

Ejemplo basado en Netconf, proceso similar para jti, snmp, openconf

ADAPTACIONES

HOSTS

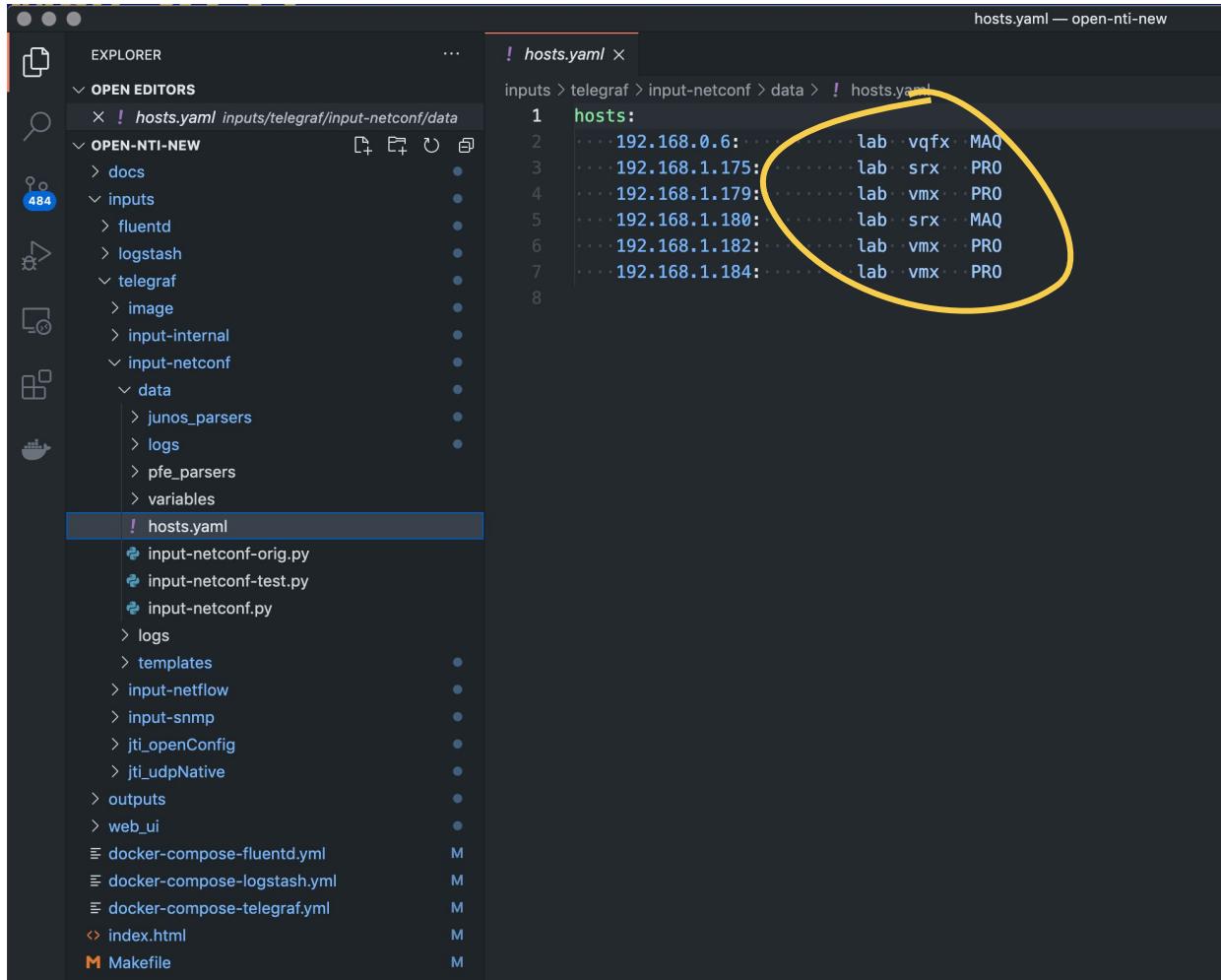


```
hosts.yaml — open-nti-new
inputs > telegraf > input-netconf > data > ! hosts.yaml
1 hosts:
2   192.168.0.6: ..... lab vqfx MAQ
3   192.168.1.175: ..... lab srx PRO
4   192.168.1.179: ..... lab vmx PRO
5   192.168.1.180: ..... lab srx MAQ
6   192.168.1.182: ..... lab vmx PRO
7   192.168.1.184: ..... lab vmx PRO
8

! hosts.yaml
! input-netconf-orig.py
! input-netconf-test.py
! input-netconf.py
> logs
> templates
> input-netflow
> input-snmp
> jti_openConfig
> jti_udpNative
> outputs
> web_ui
docker-compose-fluentd.yml M
docker-compose-logstash.yml M
docker-compose-telemetry.yml M
index.html M
Makefile M
```

ADAPTACIONES

HOSTS



The screenshot shows the VS Code interface with the following details:

- Explorer View:** Shows the project structure under "OPEN EDITORS".
 - "! hosts.yaml" is the active editor.
 - Other files listed include: input-netconf-orig.py, input-netconf-test.py, input-netconf.py, logs, templates, input-netflow, input-snmp, jti_openConfig, jti_udpNative, outputs, web_ui, docker-compose-fluentd.yml, docker-compose-logstash.yml, docker-compose-telemetry.yml, index.html, and Makefile.
- Code Editor:** Displays the contents of the hosts.yaml file.

```
hosts:  
  - 192.168.0.6: lab vqfx MAQ  
  - 192.168.1.175: lab srx PRO  
  - 192.168.1.179: lab vmx PRO  
  - 192.168.1.180: lab srx MAQ  
  - 192.168.1.182: lab vmx PRO  
  - 192.168.1.184: lab vmx PRO
```

A yellow oval highlights the "hosts:" section in the code editor.

ADAPTACIONES

HOSTS

COMANDOS

```
! commands.yaml M
inputs > telegraf > input-netconf > data > variables > ! commands.yaml
21 generic_commands:
22   commands: |
23     show chassis routing-engine | display xml
24     show chassis environment | display xml
25     show pfe statistics traffic | display xml
26     show system processes extensive
27     show system buffers
28     show system statistics icmp | display xml
29     show snmp statistics | display xml
30     show system virtual-memory | display xml
31     show system processes extensive
32     show task memory | display xml
33     show route summary | display xml
34     show bgp summary | display xml
35   tags: lab
36
37 isis_commands:
38   commands: |
39     show isis statistics | display xml
40     show route summary | display xml
41   tags: srx vmx
42
43 bng_commands:
44   commands: |
45     show pppoe statistics | display xml
46     show system processes extensive threads
47     show network-access aaa radius-servers detail | display xml
48     show subscribers summary | display xml
49     show subscribers summary port | display xml
50     show system resource-monitor subscribers-limit chassis | display xml
51     show services nat pool detail | display xml
52     show subscribers summary port | display xml
53     show system resource-monitor subscribers-limit chassis | display xml
54     show services sessions utilization | display xml
55     show services nat source pool all | display xml
56     show interfaces vms-* media | display xml
57     show snmp mib walk 1.3.6.1.4.1.2636.3.64.1.1.5.1.3 ascii | display xml
58   tags: PRO
59
60 degradation:
61   commands: |
62     show chassis fabric degradation | display xml
63     show chassis fabric reachability | display xml
64     show chassis hardware models | display xml
65   tags: MAQ
```

ADAPTACIONES

HOSTS

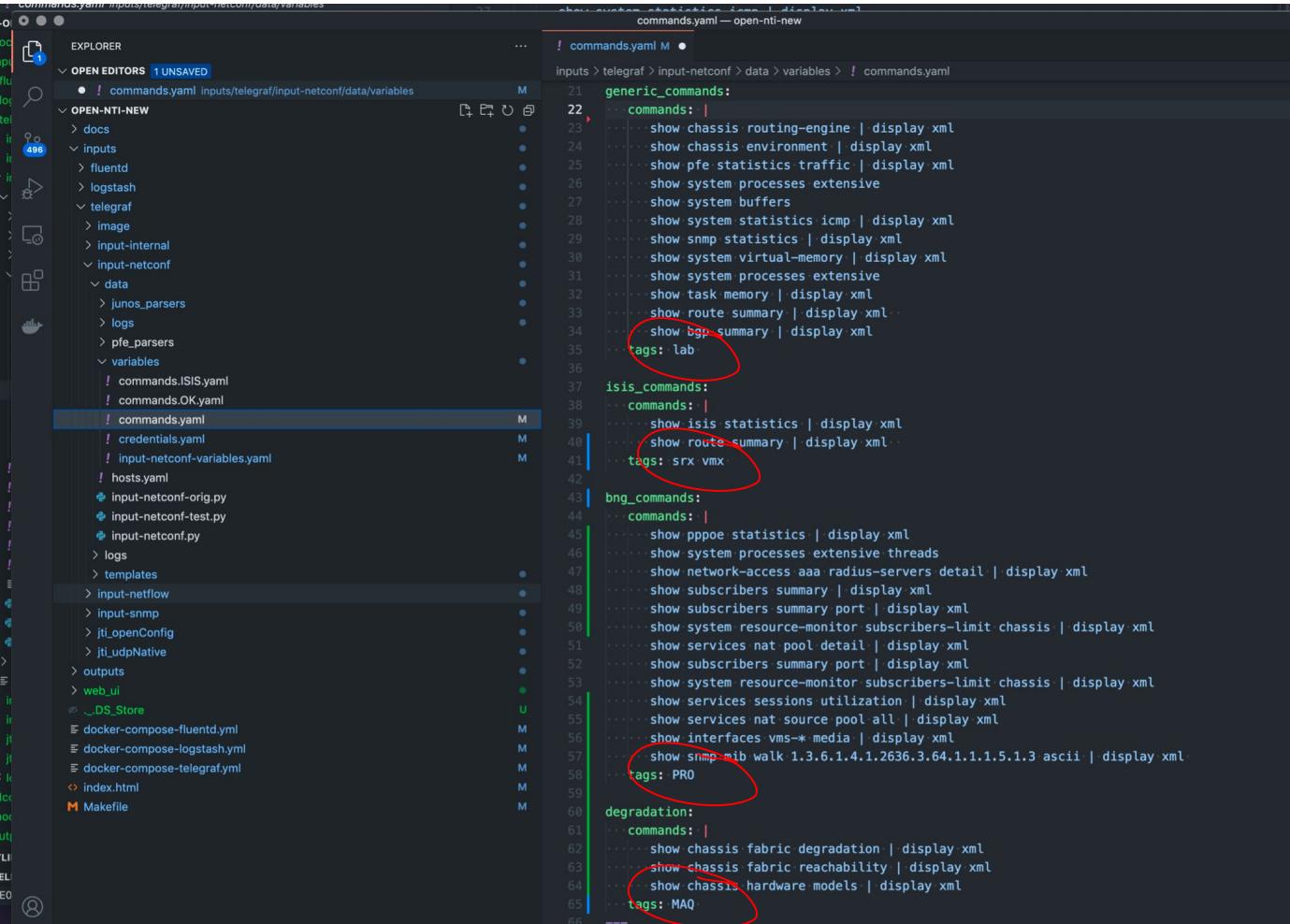
COMANDOS

```
! commands.yaml M •
inputs > telegraf > input-netconf > data > variables > ! commands.yaml
  generic_commands:
    commands: |
      show chassis routing-engine | display xml
      show chassis environment | display xml
      show pfe statistics traffic | display xml
      show system processes extensive
      show system buffers
      show system statistics icmp | display xml
      show snmp statistics | display xml
      show system virtual-memory | display xml
      show system processes extensive
      show task memory | display xml
      show route summary | display xml
      show bgp summary | display xml
      tags: lab
  isis_commands:
    commands: |
      show isis statistics | display xml
      show route summary | display xml
      tags: srx vmx
  bng_commands:
    commands: |
      show ppoe statistics | display xml
      show system processes extensive threads
      show network-access aaa radius-servers detail | display xml
      show subscribers summary | display xml
      show subscribers summary port | display xml
      show system resource-monitor subscribers-limit chassis | display xml
      show services nat pool detail | display xml
      show subscribers summary port | display xml
      show system resource-monitor subscribers-limit chassis | display xml
      show services sessions utilization | display xml
      show services nat source pool all | display xml
      show interfaces vms-media | display xml
      show snmp mib wai 1.3.6.1.4.1.2636.3.64.1.1.5.1.3 ascii | display xml
      tags: PRO
  degradation:
    commands: |
      show chassis fabric degradation | display xml
      show chassis fabric reachability | display xml
      show chassis hardware models | display xml
      tags: MAQ
```

ADAPTACIONES

HOSTS

COMANDOS



```
! commands.yaml M •
inputs > telegraf > input-netconf > data > variables > ! commands.yaml
21 generic_commands:
22   commands: |
23     show chassis routing-engine | display xml
24     show chassis environment | display xml
25     show pfe statistics traffic | display xml
26     show system processes extensive
27     show system buffers
28     show system statistics icmp | display xml
29     show snmp statistics | display xml
30     show system virtual-memory | display xml
31     show system processes extensive
32     show task memory | display xml
33     show route summary | display xml
34     show bgp summary | display xml
35     tags: lab
36
37 isis_commands:
38   commands: |
39     show isis statistics | display xml
40     show routes summary | display xml
41     tags: srx vmx
42
43 bng_commands:
44   commands: |
45     show ppoe statistics | display xml
46     show system processes extensive threads
47     show network-access aaa radius-servers detail | display xml
48     show subscribers summary | display xml
49     show subscribers summary port | display xml
50     show system resource-monitor subscribers-limit chassis | display xml
51     show services nat pool detail | display xml
52     show subscribers summary port | display xml
53     show system resource-monitor subscribers-limit chassis | display xml
54     show services sessions utilization | display xml
55     show services nat source pool all | display xml
56     show interfaces vms-* media | display xml
57     show snmp mib walk 1.3.6.1.4.1.2636.3.64.1.1.5.1.3 ascii | display xml
58     tags: PRO
59
60 degradation:
61   commands: |
62     show chassis fabric degradation | display xml
63     show chassis fabric reachability | display xml
64     show chassis hardware models | display xml
65     tags: MAQ
66
```

ADAPTACIONES

HOSTS

COMANDOS

AUTENTIFICACIÓN

The screenshot shows a dark-themed code editor interface with several windows and panels:

- Left Panel (Explorer):** Shows a file tree with the following structure:
 - OPEN EDITORS (2 UNSAVED)
 - ! commands.yaml
 - ! credentials.yaml
 - OPEN-NTI-NEW
 - ! show-subscribers-summary.parser.yaml
 - ! show-system-buffers.parser.yaml
 - ! show-system-configuration-database-usage.parser.yaml
 - ! show-system-core-dumps.parser.yaml
 - ! show-system-information.parser.yaml
 - ! show-system-information.parser.yaml.NOT_WORKING
 - ! show-system-processes-extensive.parser.yaml
 - ! show-system-resource-monitor-summary.parser.yaml
 - ! show-system-statistics-icmp.parser.yaml
 - ! show-system-storage.parser.yaml
 - ! show-system-virtual-memory.parser.yaml
 - ! show-task-accounting.parser.yaml
 - ! show-task-io.parser.yaml
 - ! show-task-memory.parser.yaml
 - ! show-version.parser.yaml
 - variables
 - ! commands.ISIS.yaml
 - ! commands.OK.yaml
 - ! commands.yaml
 - ! credentials.yaml
 - ! input-netconf-variables.yaml
 - ! hosts.yaml
- Top Right Editor:** A file titled "commands.yaml — open-nti-new" showing YAML configuration for Telegraf inputs.

```
inputs > telegraf > input-netconf > data > variables > ! commands.yaml
10  #-----show-version|display.xml
11  #-----show-system-buffers
12  #-----show-system-statistics-icmp|display.xml
```
- Bottom Right Editor:** A file titled "credentials.yaml — open-nti-new" showing YAML configuration for credentials.

```
inputs > telegraf > input-netconf > data > variables > ! credentials.yaml
1  ! lab_credentials:
2    username: juampe
3    password: pimp01
4    community: publico1
5    tags: lab
6
7  ! lob_credentials:
8    username: lab
9    password: lab123
10   community: comlrima
11   tags: lob
```
- Bottom Left Status Bar:** Shows the current branch ("master*"), file count ("0"), and tags ("MAQ").

ADAPTACIONES

HOSTS

COMANDOS

AUTENTIFICACIÓN

The screenshot displays the Visual Studio Code interface with the following details:

- Explorer View:** Shows a tree structure of files. Under "OPEN-NTI-NEW", there are numerous parser files like "show-system-information.parser.yaml", "show-system-processes-extensive.parser.yaml", etc., and several configuration files including "commands.yaml", "credentials.yaml", "input-netconf-variables.yaml", and "hosts.yaml".
- Editor View:** Contains two open files:
 - commands.yaml**: A file under "inputs > telegraf > input-netconf > data > variables". It includes lines 10, 11, and 12 which are circled in yellow.
 - credentials.yaml**: A file under "inputs > telegraf > input-netconf > data > variables". It contains two sections: "lab_credentials" and "lob_credentials". Lines 4, 11, and 12 are circled in yellow.
- Status Bar:** Shows the current file is "commands.yaml — open-nti-new". It also displays "tags: MAQ" and the commit status "master*".

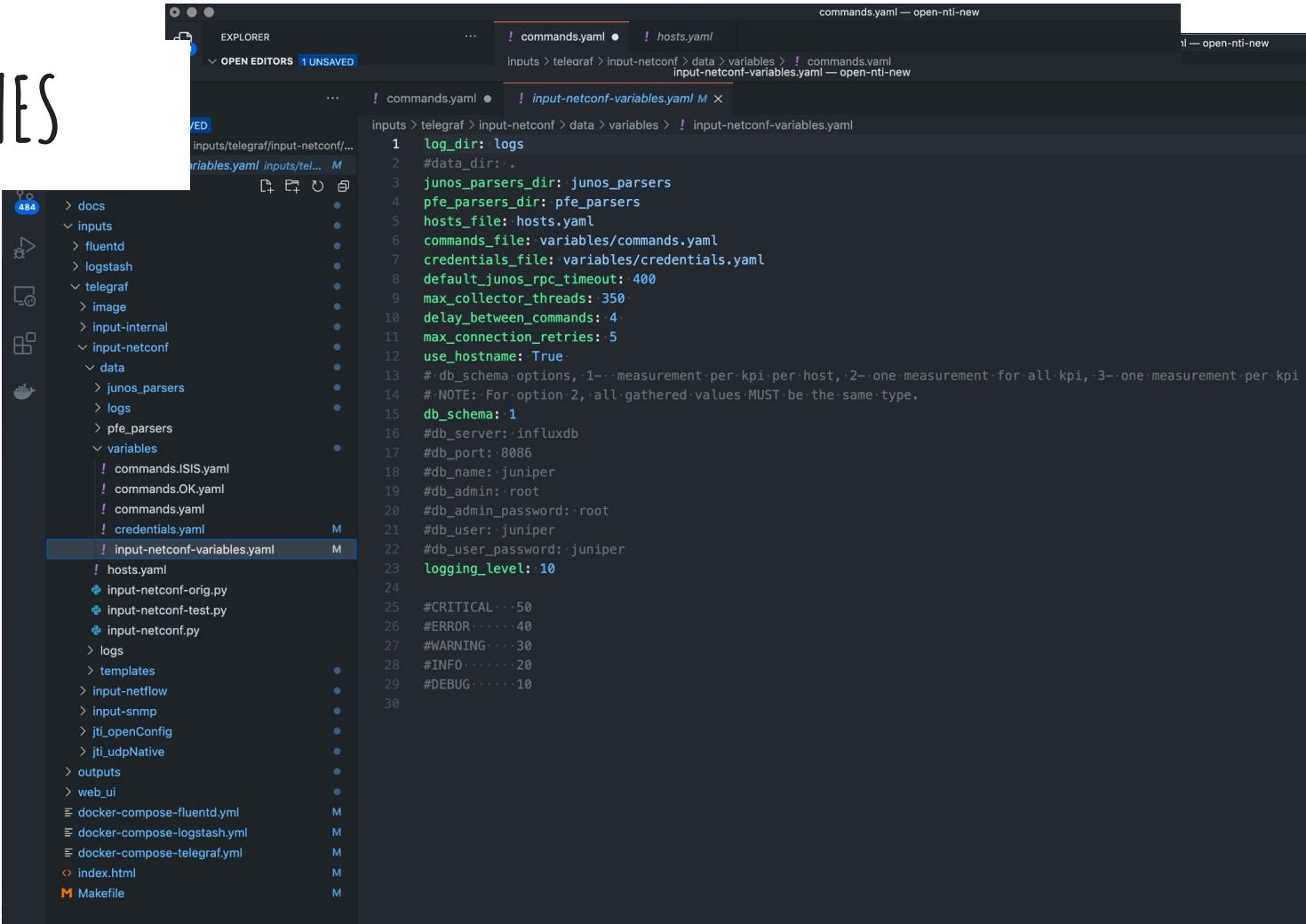
ADAPTACIONES

HOSTS

COMANDOS

AUTENTIFICACIÓN

EJECUCIÓN



```
! commands.yaml • ! hosts.yaml
inputs > telegraf > input-netconf > data > variables > ! commands.yaml
input-netconf-variables.yaml — open-nti-new

! commands.yaml • ! input-netconf-variables.yaml M ×
inputs > telegraf > input-netconf > data > variables > ! input-netconf-variables.yaml
1 log_dir: logs
2 #data_dir: .
3 junos_parsers_dir: junos_parsers
4 pfe_parsers_dir: pfe_parsers
5 hosts_file: hosts.yaml
6 commands_file: variables/commands.yaml
7 credentials_file: variables/credentials.yaml
8 default_junos_rpc_timeout: 400
9 max_collector_threads: 350
10 delay_between_commands: 4
11 max_connection_retries: 5
12 use_hostname: True
13 #db_schema: options, 1--one measurement per kpi per host, 2--one measurement for all kpi, 3--one measurement per kpi
14 #NOTE: For option 2, all gathered values MUST be the same type.
15 db_schema: 1
16 #db_server: influxdb
17 #db_port: 8086
18 #db_name: juniper
19 #db_admin: root
20 #db_admin_password: root
21 #db_user: juniper
22 #db_user_password: juniper
23 logging_level: 10
24
25 #CRITICAL...50
26 #ERROR....40
27 #WARNING....30
28 #INFO.....20
29 #DEBUG.....10
30
```

ADAPTACIONES

HOSTS

COMANDOS

AUTENTIFICACIÓN

EJECUCIÓN

The screenshot shows a terminal window with the following file structure and content:

```
inputs > telegraf > input-netconf > data > variables > ! input-netconf-variables.yaml
```

The content of the file is:

```
! input-netconf-variables.yaml M
1 log_dir: logs
2 #data_dir: .
3 junos_parsers_dir: junos_parsers
4 pfe_parsers_dir: pfe_parsers
5 hosts_file: hosts.yaml
6 commands_file: variables/commands.yaml
7 credentials_file: variables/credentials.yaml
8 default_junos_rpc_timeout: 400
9 max_collector_threads: 350
10 delay_between_commands: 4
11 max_connection_retries: 5
12 use_hostname: True
13 #db_schema_options: 1--measurement per kpi per host, 2--one measurement for all kpi, 3--one measurement per kpi
14 #NOTE: For option 2, all gathered values MUST be the same type.
15 db_schema: 1
16 #db_server: influxdb
17 #db_port: 8086
18 #db_name: juniper
19 #db_admin: root
20 #db_admin_password: root
21 #db_user: juniper
22 #db_user_password: juniper
23 logging_level: 10
24 #CRITICAL...50
25 #ERROR....40
26 #WARNING....30
27 #INFO.....20
28 #DEBUG.....10
29
30
```

Two sections of the code are circled in yellow:

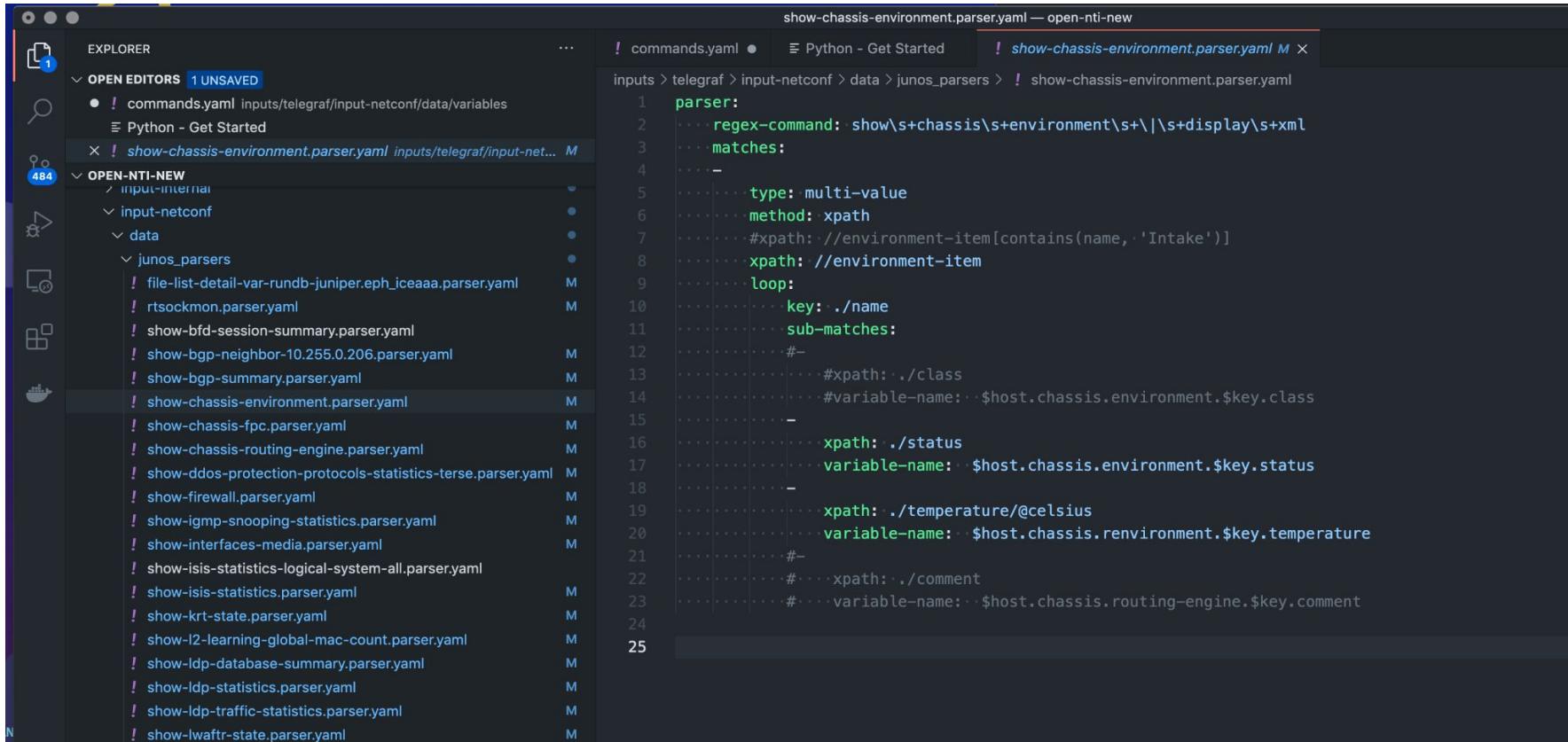
- The first circle highlights the configuration for the database connection, specifically the `db_schema` setting and its associated options.
- The second circle highlights the `logging_level` setting, which is set to 10.

ADAPTACIONES

Personalización de los KPIs a través de
PARSERS

ADAPTACIONES

Personalización de los KPIs a través de PARSERS



The screenshot shows a code editor interface with the following details:

- EXPLORER** sidebar: Shows the project structure with several files listed under "OPEN EDITORS" and "OPEN-NTI-NEW".
- COMMANDS**:
 - commands.yaml
 - inputs/telegraf/input-netconf/data/variables
 - Python - Get Started
- OPEN EDITORS**:
 - commands.yaml
 - inputs/telegraf/input-netconf/data/junos_parsers/show-chassis-environment.parser.yaml
 - Python - Get Started
- OPEN-NTI-NEW**:
 - /input-internal
 - /input-netconf
 - /data/junos_parsers
 - file-list-detail-var-rundb-juniper.eph_iceaaa.parser.yaml
 - rtsockmon.parser.yaml
 - show-bfd-session-summary.parser.yaml
 - show-bgp-neighbor-10.255.0.206.parser.yaml
 - show-bgp-summary.parser.yaml
 - show-chassis-environment.parser.yaml
 - show-chassis-fpc.parser.yaml
 - show-chassis-routing-engine.parser.yaml
 - show-ddos-protection-protocols-statistics-terse.parser.yaml
 - show-firewall.parser.yaml
 - show-igmp-snoping-statistics.parser.yaml
 - show-interfaces-media.parser.yaml
 - show-isis-statistics-logical-system-all.parser.yaml
 - show-isis-statistics.parser.yaml
 - show-krt-state.parser.yaml
 - show-l2-learning-global-mac-count.parser.yaml
 - show-ldp-database-summary.parser.yaml
 - show-ldp-statistics.parser.yaml
 - show-ldp-traffic-statistics.parser.yaml
 - show-lwaftr-state.parser.yaml

- EDITOR TABS**:
- commands.yaml
- Python - Get Started
- show-chassis-environment.parser.yaml (active tab)
- Content of show-chassis-environment.parser.yaml**:

```
parser:  
  regex-command: show\s+chassis\s+environment\s+|\s+display\s+xml  
  matches:  
    -  
      type: multi-value  
      method: xpath  
      #xpath: //environment-item[contains(name, 'Intake')]  
      xpath: //environment-item  
      loop:  
        key: ./name  
        sub-matches:  
          #-  
          #xpath: ./class  
          #variable-name: $host.chassis.environment.$key.class  
          -  
            xpath: ./status  
            variable-name: $host.chassis.environment.$key.status  
          -  
            xpath: ./temperature/@celsius  
            variable-name: $host.chassis.environment.$key.temperature  
          #-  
          #xpath: ./comment  
          #variable-name: $host.chassis.routing-engine.$key.comment
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

The screenshot shows a code editor interface with two main panes. The left pane, titled 'EXPLORER', displays a file tree under 'OPEN EDITORS'. It includes files like 'commands.yaml', 'Python - Get Started', and several parser files under 'OPEN-NTI-NEW/input-netconf/junos_parsers'. Two specific files are circled in yellow: 'show-chassis-environment.parser.yaml' and 'show-chassis-fpc.parser.yaml'. The right pane shows the content of the selected file, 'show-chassis-environment.parser.yaml'. The code defines a 'parser' block with a 'regex-command' of 'show\s+chassis\s+environment\s+|\s+display\s+xml' and a 'matches' block. It uses XPath to extract data from the command output, including 'key' and 'variable-name' assignments for various chassis parameters like status, temperature, and comments.

```
! commands.yaml • Python - Get Started ! show-chassis-environment.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-chassis-environment.parser.yaml
1 parser:
2   . . . regex-command: show\s+chassis\s+environment\s+|\s+display\s+xml
3   . . . matches:
4   . . .
5   . . . type: multi-value
6   . . . method: xpath
7   . . . #xpath: //environment-item[contains(name, 'Intake')]
8   . . . xpath: //environment-item
9   . . . loop:
10  . . .   key: ./name
11  . . .   sub-matches:
12  . . .     #-
13  . . .     #xpath: ./class
14  . . .     #variable-name: $host.chassis.environment.$key.class
15  . . .
16  . . .   xpath: ./status
17  . . .   variable-name: $host.chassis.environment.$key.status
18  . . .
19  . . .   xpath: ./temperature@celsius
20  . . .   variable-name: $host.chassis.environment.$key.temperature
21  . . .
22  . . .   #-
23  . . .   #xpath: ./comment
24  . . .   #variable-name: $host.chassis.routing-engine.$key.comment
25
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

The screenshot shows a code editor interface with the following details:

- EXPLORER:** Shows a tree view of files. A yellow oval highlights the **input-netconf** folder, which contains a **junos_parsers** folder. Inside **junos_parsers**, two files are circled in yellow: **show-chassis-environment.parser.yaml** and **show-chassis-fpc.parser.yaml**.
- OPEN EDITORS:** Shows three open files:
 - commands.yaml**: A Python configuration file.
 - Python - Get Started**: A Python template.
 - ! show-chassis-environment.parser.yaml**: The configuration file for chassis environment parsers.
- EDITOR:** Displays the content of **show-chassis-environment.parser.yaml**. The code defines a parser for the command `show chassis environment`. It uses XPath to extract specific information from the XML response. Red arrows point to several key XPath expressions:
 - `#xpath: //environment-item[contains(name, 'Intake')]`
 - `xpath: //environment-item`
 - `key: ./name`
 - `xpath: ./status`
 - `variable-name: $host.chassis.environment.$key.status`
 - `xpath: ./temperature/@celsius`
 - `variable-name: $host.chassis.environment.$key.temperature`

```
! commands.yaml • ┌ Python - Get Started ┌ ! show-chassis-environment.parser.yaml M ×
inputs > telegraf > input-netconf > data > junos_parsers > ! show-chassis-environment.parser.yaml
1   parser:
2     ...
3       regex-command: show\s+chassis\s+environment\s+|\s+display\s+xml
4       matches:
5         ...
6           type: multi-value
7           method: xpath
8           #xpath: //environment-item[contains(name, 'Intake')]
9           xpath: //environment-item
10          loop:
11              key: ./name
12              sub-matches:
13                ...
14                  #xpath: ./class
15                  ...
16                  xpath: ./status
17                  variable-name: $host.chassis.environment.$key.status
18                  ...
19                  xpath: ./temperature/@celsius
20                  variable-name: $host.chassis.environment.$key.temperature
21                  ...
22                  #...xpath: ./comment
23                  ...
24                  #...variable-name: $host.chassis.routing-engine.$key.comment
25
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

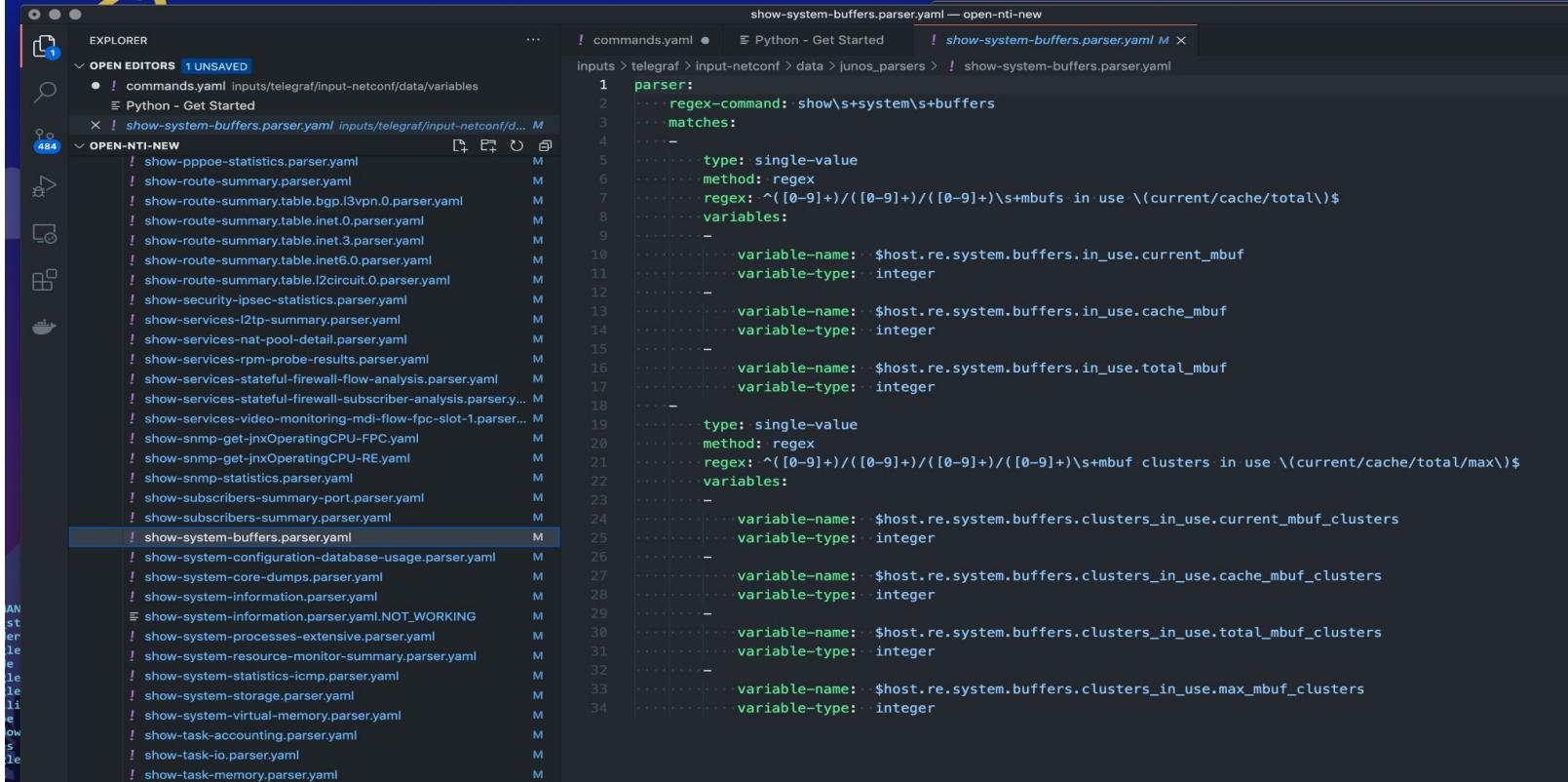
The screenshot shows a code editor interface with the following details:

- EXPLORER:** Shows several files under "OPEN EDITORS" and "OPEN-NTI-NEW". A yellow oval highlights the "show-chassis-environment.parser.yaml" file in the "OPEN-NTI-NEW" section.
- EDITOR:** The active file is "show-chassis-environment.parser.yaml".
 - Parser Definition:** The code defines a parser for the command "show chassis environment". It uses regular expressions to match the command and XPath expressions to extract data from the XML response.
 - Variable Bindings:** The code maps extracted data to variables:
 - For "status": variable-name is \$host.chassis.environment.\$key.status
 - For "temperature": variable-name is \$host.chassis.environment.\$key.temperature
 - For "comment": variable-name is \$host.chassis.routing-engine.\$key.comment

```
! commands.yaml • Python - Get Started ! show-chassis-environment.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-chassis-environment.parser.yaml
1 parser:
2   . . . regex-command: show\s+chassis\s+environment\s+|\s+display\s+xml
3   . . . matches:
4   . . .
5   . . . type: multi-value
6   . . . method: xpath
7   . . . #xpath://environment-item[contains(name,'Intake')]
8   . . . xpath: //environment-item
9   . . . loop:
10  . . .   key: ./name
11  . . .   sub-matches:
12  . . .     #-
13  . . .     #xpath: ./class
14  . . .     #variable-name: $host.chassis.environment.$key.class
15  . . .
16  . . .     xpath: ./status
17  . . .     variable-name: $host.chassis.environment.$key.status
18  . . .
19  . . .     xpath: ./temperature@celsius
20  . . .     variable-name: $host.chassis.environment.$key.temperature
21  . . .
22  . . .     #-
23  . . .     #xpath: ./comment
24  . . .     #variable-name: $host.chassis.routing-engine.$key.comment
25
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

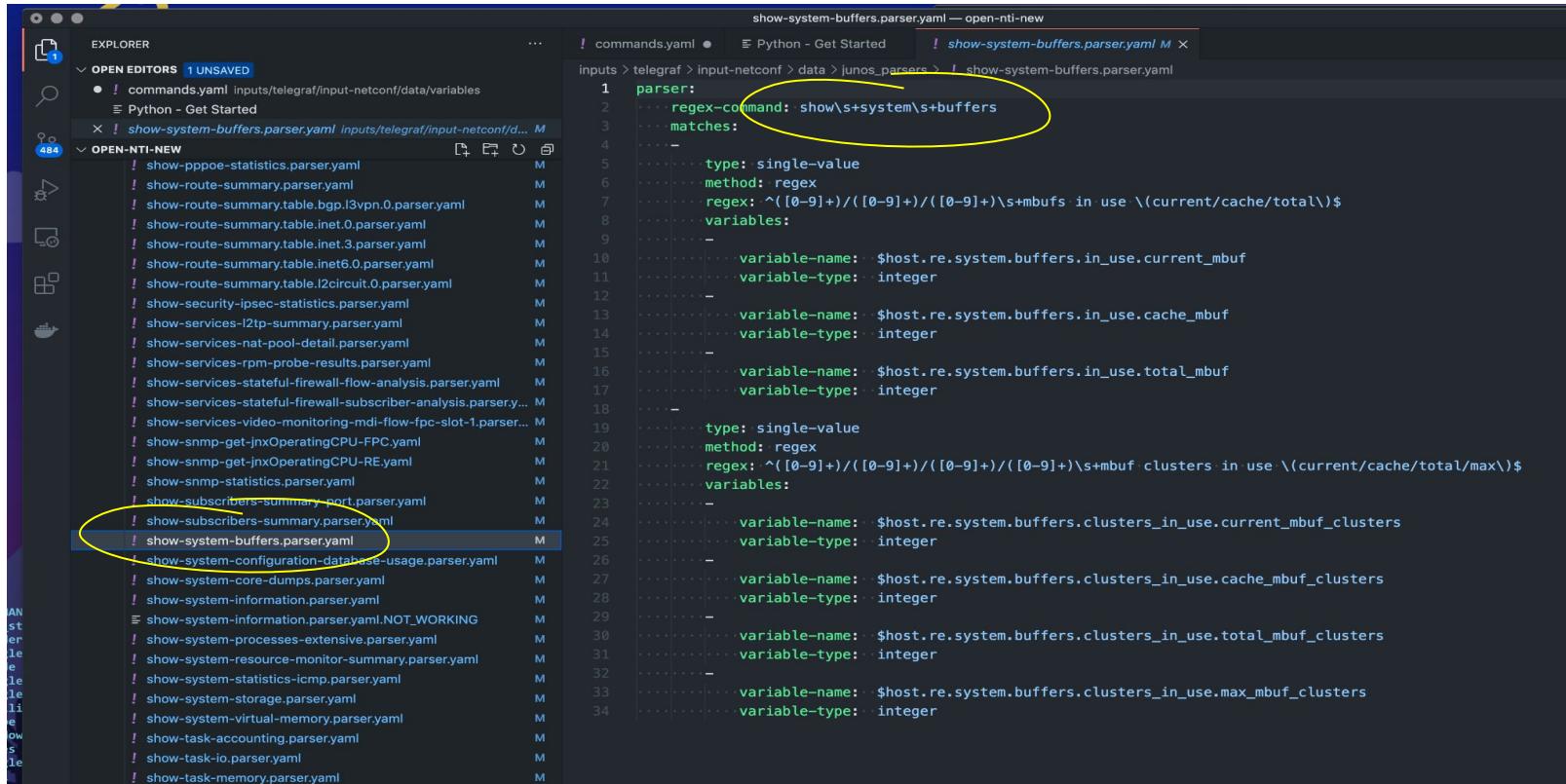


The screenshot shows a code editor interface with a sidebar on the left containing file navigation and search tools. The main area displays a YAML configuration file for a Juniper Junos parser.

```
! commands.yaml ● E Python - Get Started ! show-system-buffers.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-system-buffers.parser.yaml
1 parser:
2   . . . .
3   matches:
4     -
5       type: single-value
6       method: regex
7       regex: ^([0-9]+) / ([0-9]+) / ([0-9]+) \s+mbufs in use \((current/cache/total)\)$
8       variables:
9         -
10           variable-name: $host.re.system.buffers.in_use.current_mbuf
11           variable-type: integer
12         -
13           variable-name: $host.re.system.buffers.in_use.cache_mbuf
14           variable-type: integer
15         -
16           variable-name: $host.re.system.buffers.in_use.total_mbuf
17           variable-type: integer
18         -
19           type: single-value
20           method: regex
21           regex: ^([0-9]+) / ([0-9]+) / ([0-9]+) \s+mbuf clusters in use \((current/cache/total/max)\)$
22           variables:
23             -
24               variable-name: $host.re.system.buffers.clusters_in_use.current_mbuf_clusters
25               variable-type: integer
26             -
27               variable-name: $host.re.system.buffers.clusters_in_use.cache_mbuf_clusters
28               variable-type: integer
29             -
30               variable-name: $host.re.system.buffers.clusters_in_use.total_mbuf_clusters
31               variable-type: integer
32             -
33               variable-name: $host.re.system.buffers.clusters_in_use.max_mbuf_clusters
34               variable-type: integer
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS



The screenshot shows a code editor interface with a left sidebar labeled "EXPLORER" containing a list of files under "OPEN EDITORS" and "OPEN-NTI-NEW". A yellow circle highlights the "show-system-buffers.parser.yaml" file in the "OPEN-NTI-NEW" section. Another yellow circle highlights the "parser:" block in the code editor window.

```
! commands.yaml ● E Python - Get Started ! show-system-buffers.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-system-buffers.parser.yaml
1 parser:
2   - regex-command: show\s+system\s+buffers
3     matches:
4       -
5         type: single-value
6         method: regex
7         regex: ^([0-9]+)/([0-9]+)/([0-9]+)\$mbufs in use \((current/cache/total)\$
8         variables:
9           -
10             variable-name: $host.re.system.buffers.in_use.current_mbuf
11             variable-type: integer
12           -
13             variable-name: $host.re.system.buffers.in_use.cache_mbuf
14             variable-type: integer
15           -
16             variable-name: $host.re.system.buffers.in_use.total_mbuf
17             variable-type: integer
18           -
19             type: single-value
20             method: regex
21             regex: ^([0-9]+)/([0-9]+)/([0-9]+)\$mbuf clusters in use \((current/cache/total/max)\$
22             variables:
23               -
24                 variable-name: $host.re.system.buffers.clusters_in_use.current_mbuf_clusters
25                 variable-type: integer
26               -
27                 variable-name: $host.re.system.buffers.clusters_in_use.cache_mbuf_clusters
28                 variable-type: integer
29               -
30                 variable-name: $host.re.system.buffers.clusters_in_use.total_mbuf_clusters
31                 variable-type: integer
32               -
33                 variable-name: $host.re.system.buffers.clusters_in_use.max_mbuf_clusters
34                 variable-type: integer
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

The screenshot shows a code editor interface with the following details:

- EXPLORER:** Shows a list of files under "OPEN EDITORS" and "OPEN-NTI-NEW". The file "show-system-buffers.parser.yaml" is highlighted with a yellow circle.
- EDITOR:** Displays the content of "show-system-buffers.parser.yaml".

```
! commands.yaml ● E Python - Get Started ! show-system-buffers.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-system-buffers.parser.yaml
1 parser:
2   - regex-command: show\s+system\s+buffers
3     matches:
4       -
5         type: single-value
6         method: regex
7         regex: '^(0-9+)/([0-9+)/([0-9+)\$+mbufs in use \((current/cache/total\)$
8         variables:
9           -
10             variable-name: $host.re.system.buffers.in_use.current_mbuf
11             variable-type: integer
12           -
13             variable-name: $host.re.system.buffers.in_use.cache_mbuf
14             variable-type: integer
15           -
16             variable-name: $host.re.system.buffers.in_use.total_mbuf
17             variable-type: integer
18           -
19             type: single-value
20             method: regex
21             regex: '^(0-9+)/([0-9+)/([0-9+)\$+mbuf clusters in use \((current/cache/total/max\)$
22             variables:
23               -
24                 variable-name: $host.re.system.buffers.clusters_in_use.current_mbuf_clusters
25                 variable-type: integer
26               -
27                 variable-name: $host.re.system.buffers.clusters_in_use.cache_mbuf_clusters
28                 variable-type: integer
29               -
30                 variable-name: $host.re.system.buffers.clusters_in_use.total_mbuf_clusters
31                 variable-type: integer
32               -
33                 variable-name: $host.re.system.buffers.clusters_in_use.max_mbuf_clusters
34                 variable-type: integer
```

ADAPTACIONES

Personalización de los KPIs a través de PARSERS

```
! commands.yaml ● E Python - Get Started ! show-system-buffers.parser.yaml M
inputs > telegraf > input-netconf > data > junos_parsers > ! show-system-buffers.parser.yaml
1 parser:
2   - regex-command: show\s+system\s+buffers
3     matches:
4       -
5         type: single-value
6         method: regex
7         regex: ^([0-9]+)/([0-9]+)/([0-9]+)\$mbufs in use \((current/cache/total)\$
8         variables:
9           -
10             variable-name: $host.re.system.buffers.in_use.current_mbuf
11             variable-type: integer
12           -
13             variable-name: $host.re.system.buffers.in_use.cache_mbuf
14             variable-type: integer
15           -
16             variable-name: $host.re.system.buffers.in_use.total_mbuf
17             variable-type: integer
18           -
19             type: single-value
20             method: regex
21             regex: ^([0-9]+)/([0-9]+)/([0-9]+)\$mbuf clusters in use \((current/cache/total/max)\$
22             variables:
23               -
24                 variable-name: $host.re.system.buffers.clusters_in_use.current_mbuf_clusters
25                 variable-type: integer
26               -
27                 variable-name: $host.re.system.buffers.clusters_in_use.cache_mbuf_clusters
28                 variable-type: integer
29               -
30                 variable-name: $host.re.system.buffers.clusters_in_use.total_mbuf_clusters
31                 variable-type: integer
32               -
33                 variable-name: $host.re.system.buffers.clusters_in_use.max_mbuf_clusters
34                 variable-type: integer
```

ADAPTACIONES

Personalización del setup Docker

ADAPTACIONES

Personalización del setup Docker

Entorno docker-compose por cada configuración, 3 preconfigurados:

docker-compose-fluentd.yml
docker-compose-logstash.yml
docker-compose-tegraf.yml

Especificar:

1. **Colectores** a utilizar (*telegraf*)
2. **DB y herramientas** (*influxdb, chronograph, karabiner*)
3. **GUI** (*grafana*)
4. **Otros** (*grafana-reporting*)

(ver editor en vivo)

ARRANQUE

Operación vía docker/docker-compose

- Primer arranque: descarga de imágenes, creacion de containers

ARRANQUE

Operación vía docker/docker-compose

- Primer arranque: descarga de imagenes, creacion de containers
- Arranque parada controlada (destrucción de containers sin almacenamiento persistente)

```
root@spartacus3:~/open-nti-new# docker-compose -f docker-compose-telegraf.yml start
Starting influxdb      ... done
Starting input-internal ... done
Starting input-netconf   ... done
Starting input-snmp     ... done
Starting input-netflow-telegraf ... done
Starting jti_openconfig_telegraf ... done
Starting jti_udp_telegaf ... done
Starting kapacitor      ... done
Starting chronograf     ... done
Starting grafana        ... done
Starting renderer       ... done
root@spartacus3:~/open-nti-new# docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
58509170c573	quay.io/influxdb/chronograf:1.5.0.1	"/usr/bin/chronograf..."	4 days ago	Up 28 seconds	0.0.0.0:8888->8888/tcp	chronograf
bb8f74f4ad1d	open-nti-new_input_internal	"/entrypoint.sh /sou..."	4 days ago	Up 29 seconds	8092/udp, 8125/udp, 8094/tcp	input-internal
e9fd6c83323a	open-nti-new_input_netflow-telegraf	"/entrypoint.sh /sou..."	4 days ago	Up 36 seconds	8092/udp, 0.0.0.0:2055->2055/udp, 8125/udp, 8094/tcp	input-netflow-telegraf
0b6a4b2d2875	open-nti-new_input_netconf	"/entrypoint.sh /sou..."	4 days ago	Up 32 seconds	8092/udp, 8125/udp, 8094/tcp	input-netconf
b9aa9120b43c	open-nti-new_jti_udp_telegaf	"/entrypoint.sh /sou..."	4 days ago	Up 30 seconds	8092/udp, 8125/udp, 8094/tcp, 0.0.0.0:50000->50000/udp	jti_udpNative_telegaf
8cd17bb48434	open-nti-new_jti_openconfig_telegaf	"/entrypoint.sh /sou..."	4 days ago	Up 34 seconds	8092/udp, 8125/udp, 8094/tcp, 0.0.0.0:50051->50051/tcp	jti_openConfig_telegaf
bc58091b19b7	open-nti-new_input_snmp	"/entrypoint.sh /sou..."	4 days ago	Up 33 seconds	8092/udp, 0.0.0.0:162->162/udp, 8125/udp, 8094/tcp	input-snmp
e7d3ec011bc2	kapacitor:1.5.0	"/entrypoint.sh kapa..."	4 days ago	Up 35 seconds	0.0.0.0:9092->9092/tcp	kapacitor
81ec61fdcb4	grafana/grafana	"/run.sh"	4 days ago	Up 46 seconds	0.0.0.0:3000->3000/tcp	grafana
2f476d89291f	grafana/grafana-image-renderer:2.0.0-beta1	"dumb-init -- node b..."	4 days ago	Up 46 seconds	0.0.0.0:8081->8081/tcp	open-nti-new_renderer_1
a67a526e0b0e	influxdb:1.6.4	"/entrypoint.sh infl..."	4 days ago	Up 45 seconds	0.0.0.0:8086->8086/tcp, 0.0.0.0:8090->8090/tcp	influxdb

ARRANQUE

Operación vía docker/docker-compose

- Primer arranque: descarga de imágenes, creación de containers
- Arranque parada controlada (destrucción de containers sin almacenamiento persistente)
- Acceso a logs, entrada en containers para resolución de problemas, etc.

```
root@spartacus3:~/open-nti-new#
root@spartacus3:~/open-nti-new# docker exec -it input-netconf bash
bash-4.3#
bash-4.3# ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   6164  1568 ?        Ss  10:34  0:00 /bin/bash /entrypoint.sh /source/start.sh
root         8  0.0  0.3  71076 39368 ?        Sl  10:34  0:00 /usr/bin/telegraf --config /source/telegraf.conf
root        66  0.3  0.0   6240  1804 pts/0    Ss  10:39  0:00 bash
root        71 15.5  0.5 177576 63612 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.0.6 --start_delay 150
root        72 16.0  0.5 177576 63660 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.1.184 --start_delay 75
root        73 16.0  0.5 177576 63612 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.1.180 --start_delay 25
root        74 17.0  0.5 177576 63716 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.1.179 --start_delay 125
root        75 16.5  0.5 177576 63796 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.1.175 --start_delay 100
root        76 16.0  0.5 177576 63612 ?        S  10:39  0:00 /usr/bin/python /source/data/input-netconf.py -s --host 192.168.1.182 --start_delay 50
root        89  0.0  0.0   5660   660 pts/0    R+  10:40  0:00 ps aux
bash-4.3#
```

rlcmittcommand.yml
% 227 24.0KB/s 00:00

ARRANQUE

```
Starting renderer ... done
root@spartacus3:~/open-nti-new# docker logs -n 100 -f input-netconf
2021-05-25T10:44:11Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:44:30Z D! [outputs.influxdb] wrote batch of 1 metrics in 30.906874ms
2021-05-25T10:44:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:44:41Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:45:00Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:45:12Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:45:30Z D! [outputs.influxdb] wrote batch of 137 metrics in 637.410407ms
2021-05-25T10:45:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:45:55Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:46:00Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:46:25Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:46:30Z D! [outputs.influxdb] wrote batch of 2 metrics in 46.029292ms
2021-05-25T10:46:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:46:55Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:47:00Z D! [outputs.influxdb] wrote batch of 1 metrics in 468.048767ms
2021-05-25T10:47:00Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:47:25Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:47:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:47:55Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:48:00Z D! [outputs.influxdb] wrote batch of 1 metrics in 76.524938ms
2021-05-25T10:48:00Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:48:30Z D! [outputs.influxdb] wrote batch of 136 metrics in 237.398402ms
2021-05-25T10:48:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:48:39Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:49:00Z D! [outputs.influxdb] wrote batch of 1 metrics in 41.693417ms
2021-05-25T10:49:00Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:49:09Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:49:30Z D! [outputs.influxdb] wrote batch of 1 metrics in 31.074736ms
2021-05-25T10:49:30Z D! [outputs.influxdb] buffer fullness: 0 / 100000 metrics.
2021-05-25T10:49:39Z W! [agent] input "inputs.exec" did not complete within its interval
2021-05-25T10:50:00Z D! [outputs.influxdb] wrote batch of 1 metrics in 232.880989ms
```

Operación vía docker/docker-compose

- Primer arranque: descarga de imágenes, creacion de containers
- Arranque parada controlada (destrucción de containers sin almacenamiento persistente)
- Acceso a logs, entrada en containers para resolución de problemas, etc.

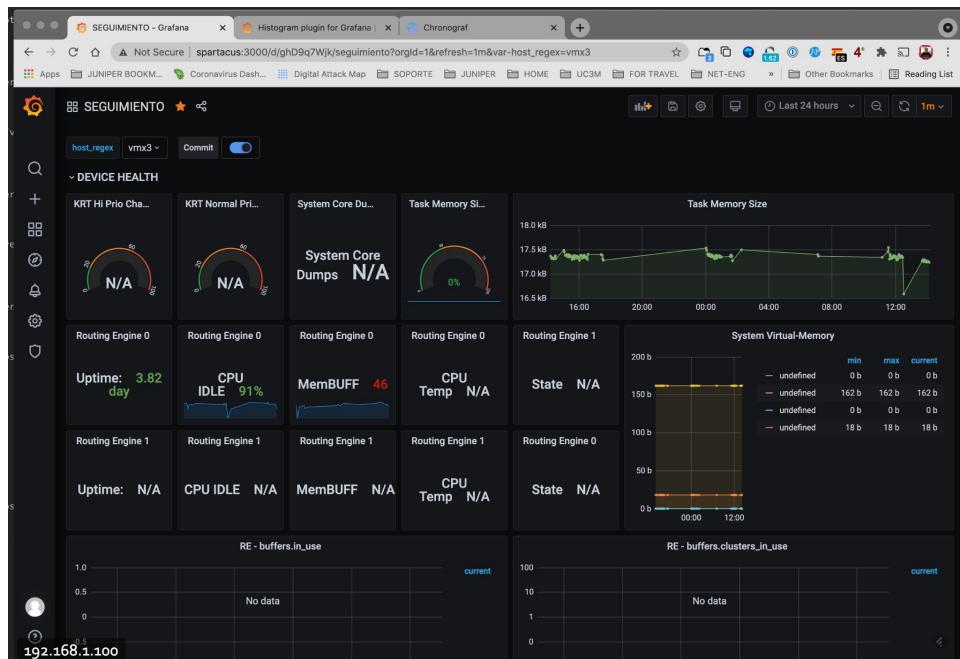
```
h /source/start.sh
fig /source/telegraf.conf
```

```
/data/input-netconf.py -s --host 192.168.0.6 --start_delay 150
/data/input-netconf.py -s --host 192.168.1.184 --start_delay 75
/data/input-netconf.py -s --host 192.168.1.180 --start_delay 25
/data/input-netconf.py -s --host 192.168.1.179 --start_delay 125
/data/input-netconf.py -s --host 192.168.1.175 --start_delay 100
/data/input-netconf.py -s --host 192.168.1.182 --start_delay 50
```

ARRANQUE

The screenshot shows the Grafana search interface. It includes a search bar at the top labeled "Search dashboards by name" and a "Sort (Default A-Z)" dropdown. Below this, there are two sections: "Starred" and "Recent". The "Starred" section contains three items: "DASHBOARD JPC" (netconf, telegraf), "HEALTH CHECK" (netconf, telegraf), and "TEMPERATURES". The "Recent" section contains several items, including "SEGUIMIENTO" (netconf, telegraf), "Data Collection Agent Dashboard" (netconf, telegraf), "JTI-OC-INTERFACES-telegraf" (OC, Jti, telegraf), "Open-NTI-SNMP" (snmp, telegraf), "JTH-UDP-FPC-CPU-telegraf" (UDP, Jti, telegraf), "JTH-UDP-INTERFACES-telegraf" (UDP, Jti, telegraf), and "JTI-QC-FPC-CPU-telegraf" (OC, Jti, telegraf). At the bottom left, there is a status indicator for the IP address 192.168.1.100.

Interfaz usuario GRAFANA



ARRANQUE

Interfaz usuario

GRAFANA y Chronograf (:8888)

The screenshot displays three windows related to monitoring and data visualization:

- Grafana Dashboard (Left Window):** Shows various dashboards under categories like DASHBOARD IPC, HEALTH CHECK, and TEMPERATURES. One dashboard is titled "SEGUIMIENTO".
- Data Explorer (Middle Window):** A query editor window showing the following SQL-like query:

```
SELECT mean("value") AS "mean_value" FROM "juniper"."four_weeks"."srx3.protocols.bgp.summary.peer-count" WHERE time > now() - 1h GROUP BY time(:interval) FILL (null)
```

The results pane shows measurements and tags for the query, including "srx3.protocols.bgp.summary.peer-count" and its sub-measurements like "device", "host", and "kpi".
- Chronograf (Right Window):** A real-time data viewer showing system memory metrics. It includes a line chart for "Task Memory Size" and a histogram for "System Virtual-Memory" with four bins labeled 0 b, 50 b, 100 b, and 150 b.

PHC BNG Hosts - Grafana

localhost:3000/d/Nq6ApoSMz/phc-bng-hosts?orgId=1

Apps SOPORTE HOME UC3M ADMINISTRIVIA FOR TRAVEL NET-ENG NET-OPS WEB-OPS DEV-OPS CODE MAC LINUX

PHC / PHC BNG Hosts

host_regex: ALCATEL-LUCENT-7810 Commit

TESTING_IPC (1 panel)

DEVICE HEALTH (19 panels) **(This panel is circled in blue)**

Routing Engine PROCESSES (4 panels)

PFE STATISTICS (6 panels)

Routing Tables Information (5 panels)

BGP STATISTICS (2 panels)

BNG Subscribers Statistics (2 panels)

BNG Sessions by type (3 panels)

BNG PPPoE Statistics (6 panels)

BNG RADIUS AUTH STATISTICS (7 panels)

BNG RADIUS ACCT STATISTICS (8 panels)

BNG POOL STATISTICS (3 panels)

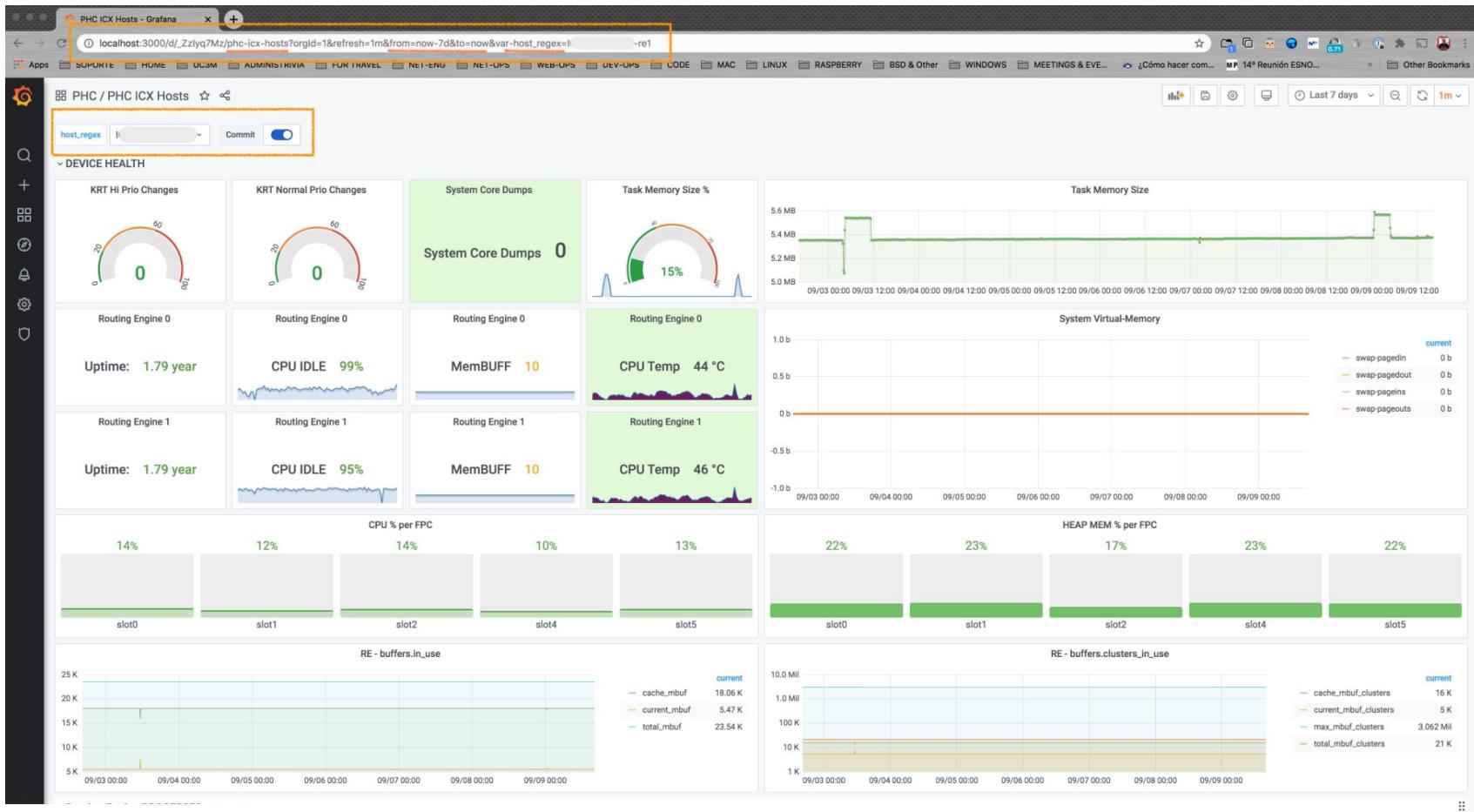
CGNAT Pools info (3 panels)

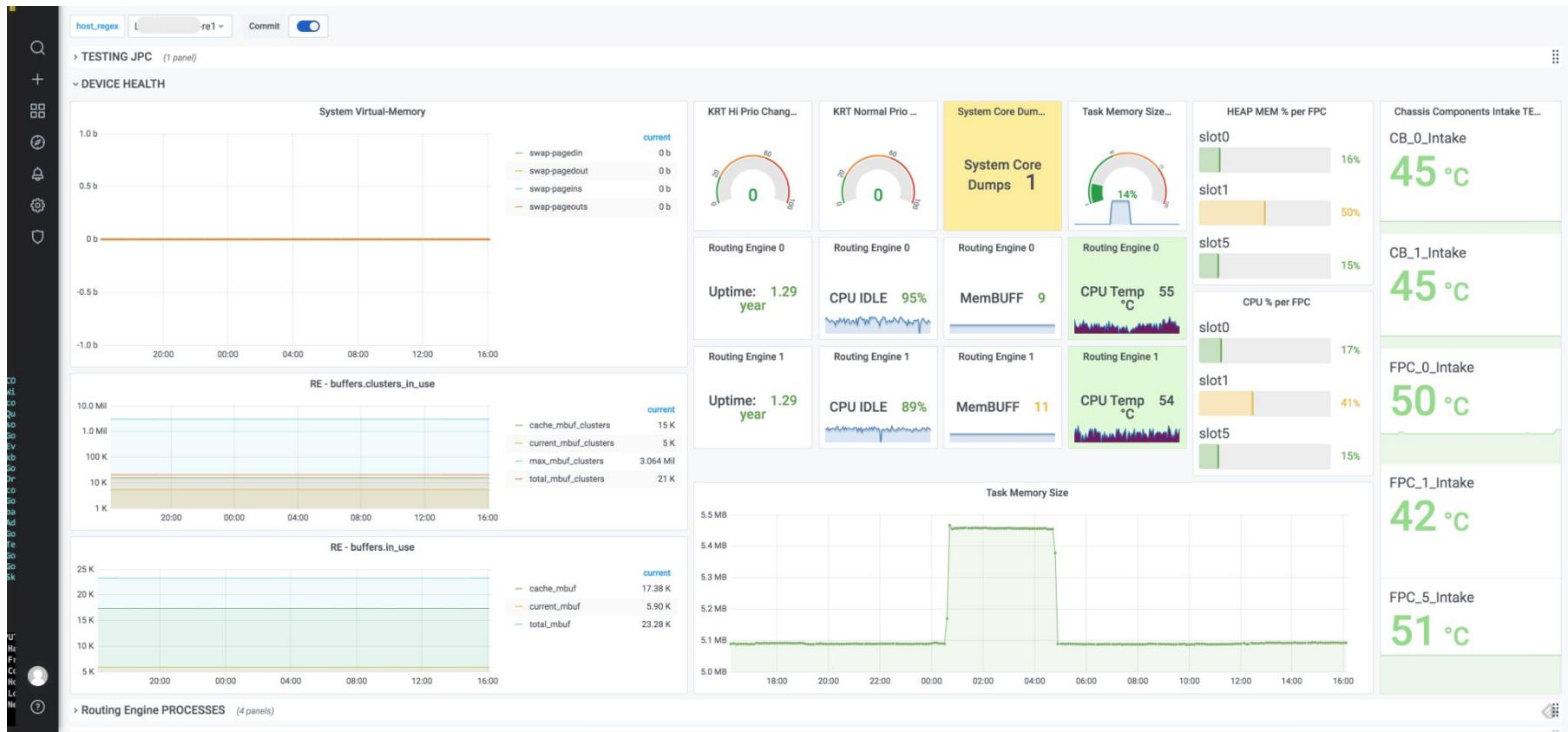
CGNAT services session info (6 panels)

PFE Resources (4 panels)

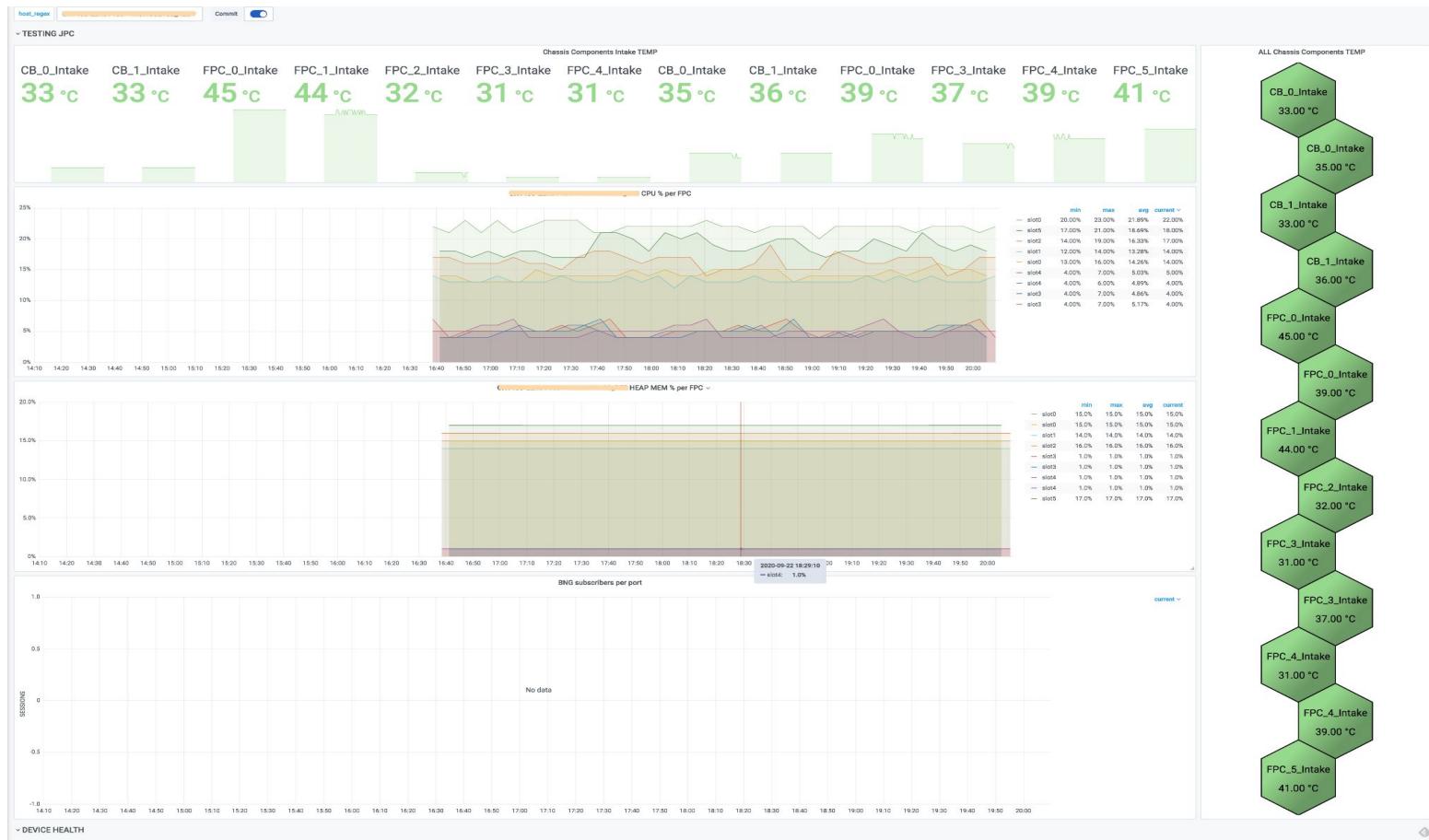
ISIS STATISTICS (8 panels)

EXPERIMENTO I - HEALTHCHECK





EXPERIMENTO I - HEALTHCHECK

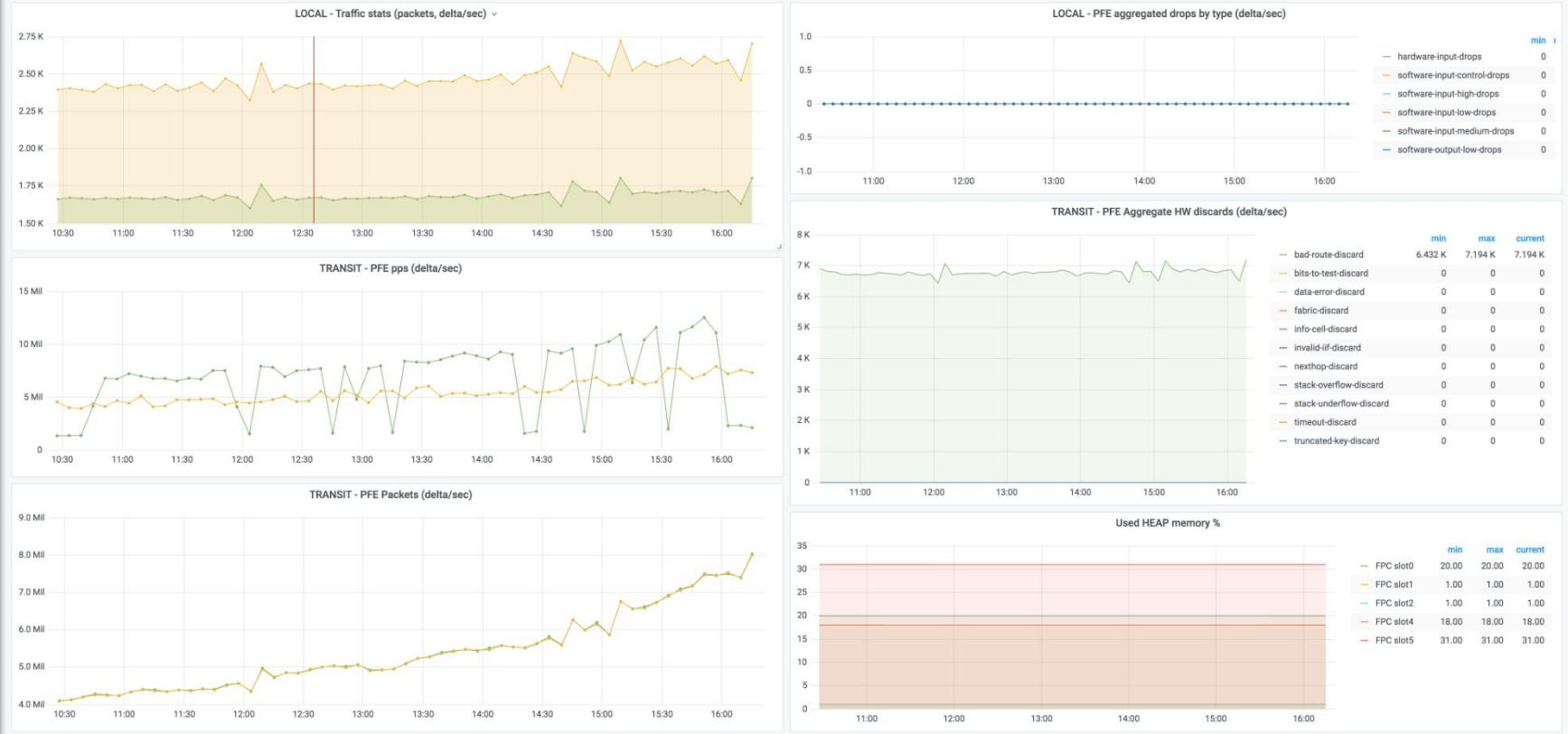


EXPERIMENTO I - HEALTHCHECK

The screenshot shows a Grafana dashboard titled "PHC / PHC BNG Hosts". The top navigation bar includes links for Apps, SOPORTE, HOME, UC3M, ADMINISTRIVIA, FOR TRAVEL, NET-ENG, NET-OPS, WEB-OPS, DEV-OPS, CODE, MAC, and LINUX. The main content area displays a list of monitoring panels:

- > TESTING JPC (1 panel)
- > DEVICE HEALTH (19 panels)
- > Routing Engines PROCESSES (4 panels)
- > PFE STATISTICS (6 panels)
- > Routing Tables Information (5 panels)
- > BGP STATISTICS (8 panels)
- > BNG Subscribers Statistics (2 panels)
- > BNG Sessions by type (3 panels)
- > BNG PPPoE Statistics (6 panels)
- > BNG RADIUS AUTH STATISTICS (7 panels)
- > BNG RADIUS ACCT STATISTICS (8 panels)
- > BNG POOL STATISTICS (3 panels)
- > CGNAT Pools info (3 panels)
- > CGNAT services session info (6 panels)
- > PFE Resources (4 panels)
- > ISIS STATISTICS (8 panels)

EXPERIMENTO II - DATA PLANE & ROUTING

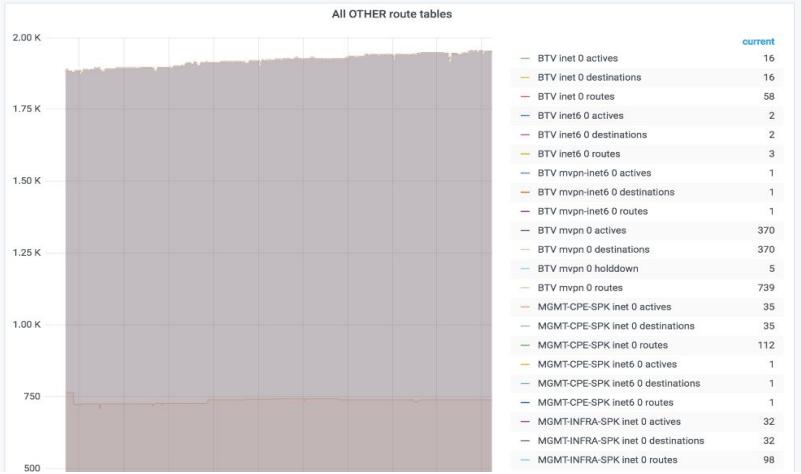
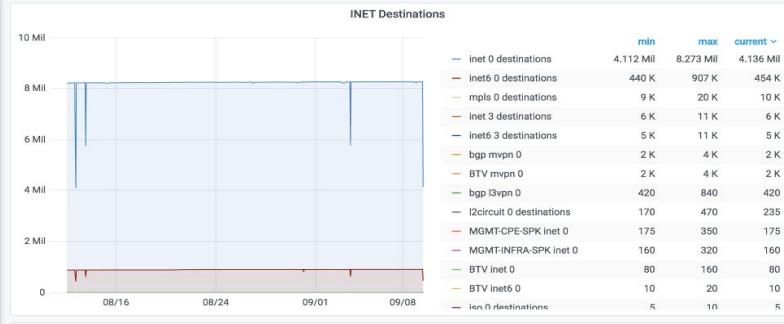
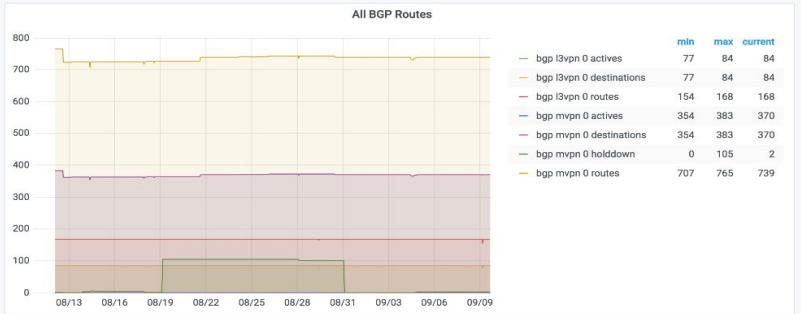
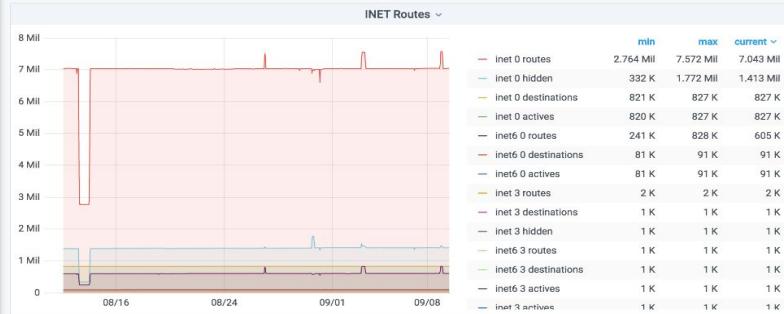


EXPERIMENTO II - DATA PLANE



PHC / PHC HEALTH CHECK mods ⚡

Routing Tables Information



EXPERIMENTO II - ROUTING

PHC BNG Hosts - Grafana

localhost:3000/d/Nq6ApoSMz/phc-bng-hosts?orgId=1

Apps SOPORTE HOME UC3M ADMINISTRIVIA FOR TRAVEL NET-ENG NET-OPS WEB-OPS DEV-OPS CODE MAC LINUX

PHC / PHC BNG Hosts

host_regex: All hosts except lo1 | Commit |

- > TESTING JPC (1 panel)
- > DEVICE HEALTH (19 panels)
- > Routing Engine PROCESSES (4 panels)
- > PFE STATISTICS (6 panels)
- > Routing Tables Information (5 panels)
- > BGP STATISTICS (2 panels)
- > BNG Subscribers Statistics (2 panels)
- > BNG Sessions by type (3 panels)
- > BNG PPPoE Statistics (6 panels)
- > BNG RADIUS AUTH STATISTICS (7 panels)
- > BNG RADIUS ACCT STATISTICS (8 panels)
- > BNG POOL STATISTICS (3 panels)
- > CGNAT Pools info (3 panels)
- > CGNAT services session info (6 panels)
- > PFE Resources (4 panels)
- > ISIS STATISTICS (8 panels)
- > RPM Probes (1 panel)

EXPERIMENTO III - BNG & CGNAT



PHC / PHC HEALTH CHECK mods ★ 🔍

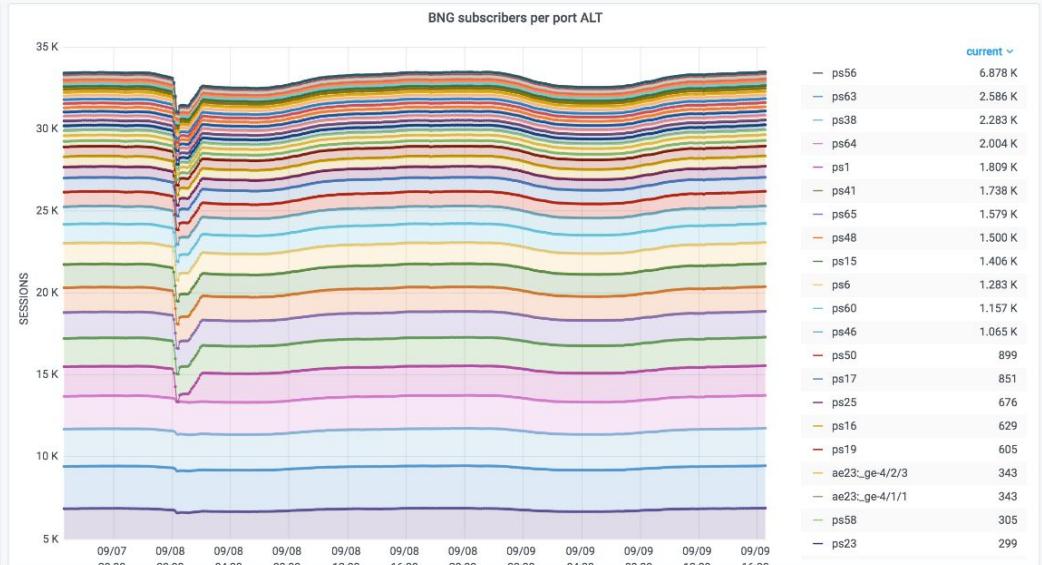
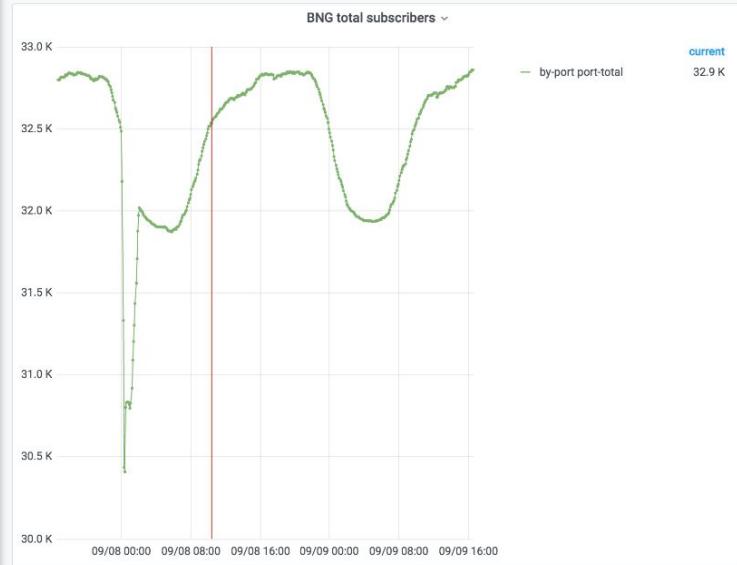
Last 2 days 1m



> Routing Tables Information (5 panels)

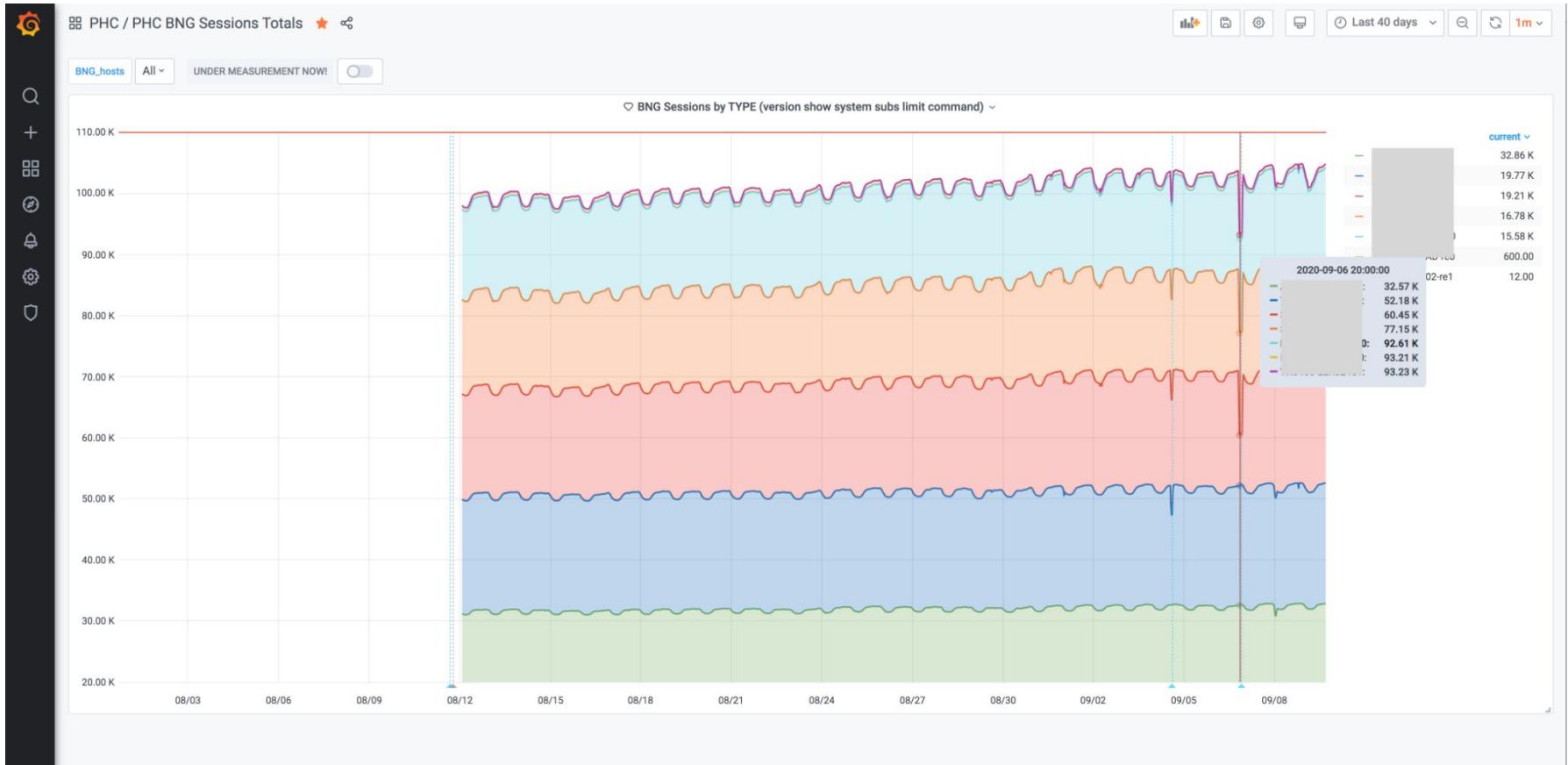
> BGP STATISTICS (2 panels)

▼ BNG Subscribers Statistics



> BNG Sessions by type (3 panels)

EXPERIMENTO III - BNG & CGNAT



EXPERIMENTO III - BNG & CGNAT



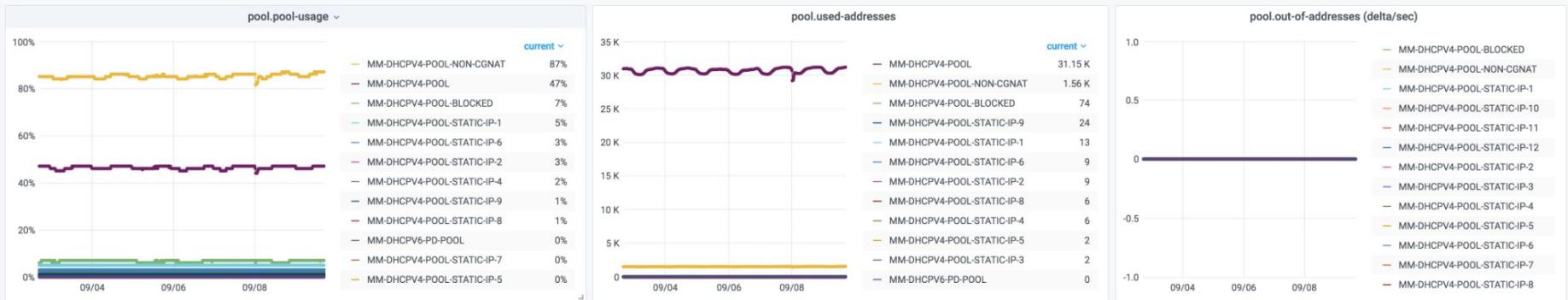
▼ BNG RADIUS AUTH STATISTICS



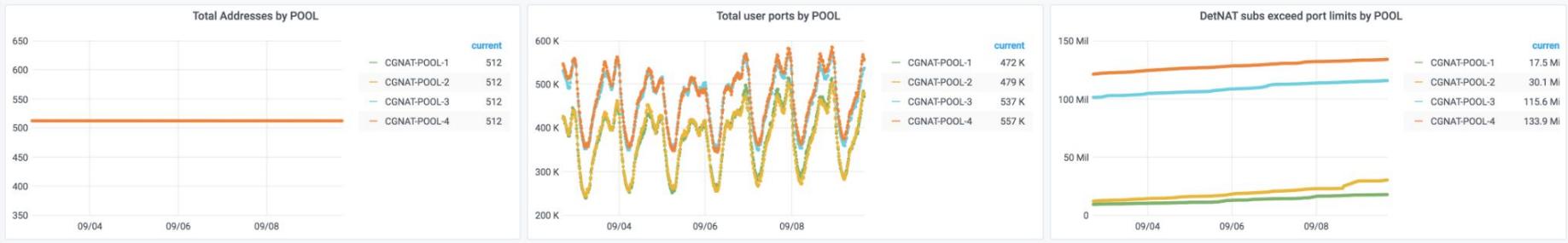
EXPERIMENTO III - BNG & CGNAT



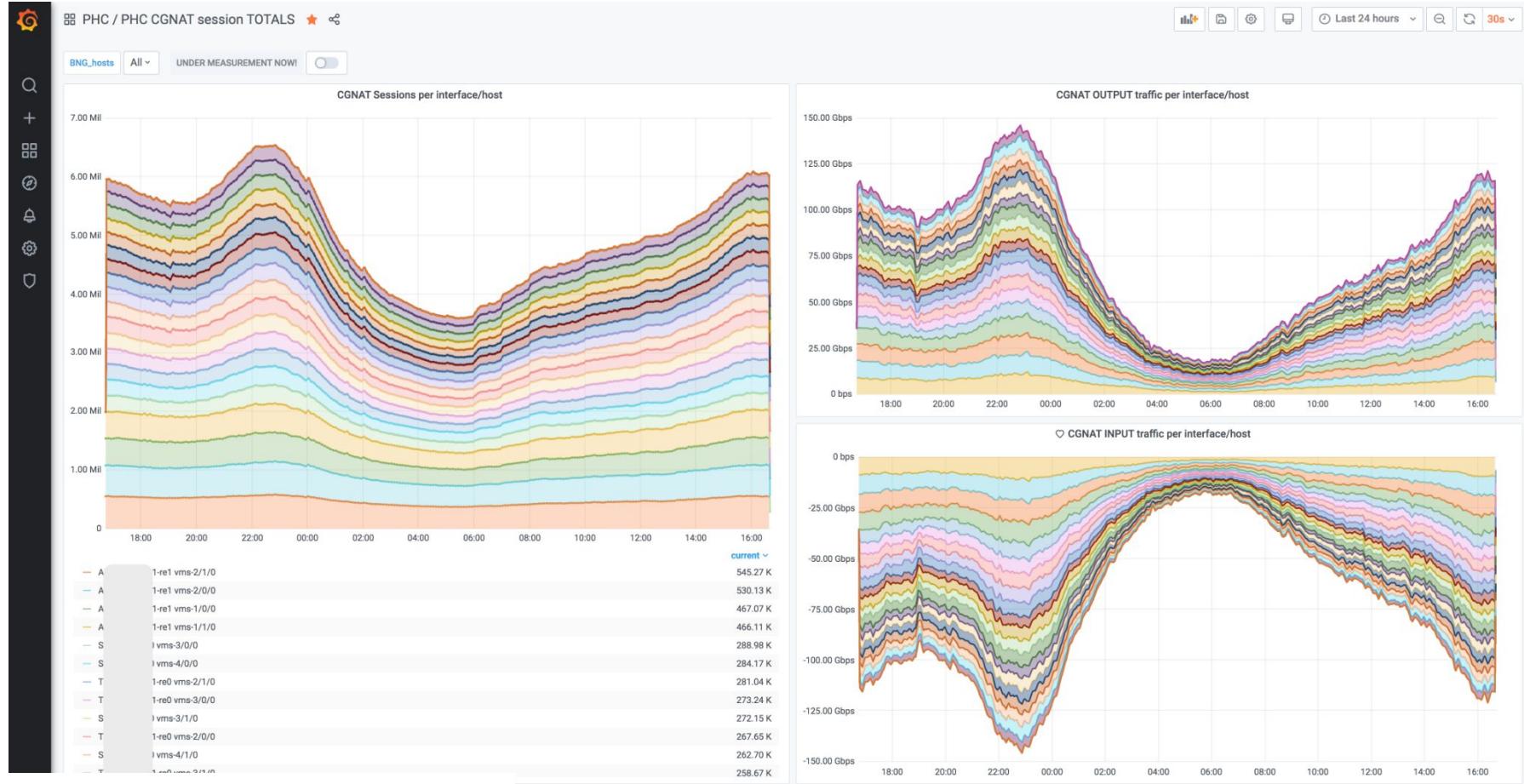
v BNG POOL STATISTICS



v CGNAT Pools info



EXPERIMENTO III - BNG & CGNAT



EXPERIMENTO III - BNG & CGNAT

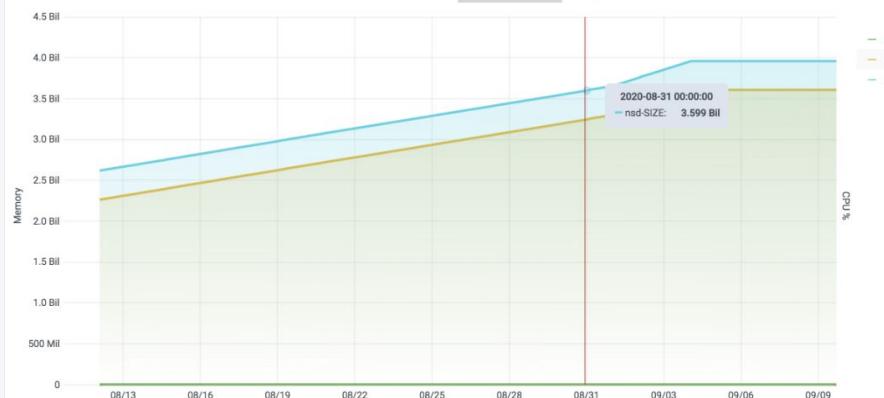


EXPERIMENTO IV - MEMORY LEAKS

UNDER MEASUREMENT NOW!



- NSD process



CPU %

- NSD RES and SIZE graphs



1 09/03 09/06 09/09

- NSD RES values

Time	mean
2020-08-12 02:00:00	2.264 GB
2020-08-12 04:00:00	2.269 GB
2020-08-12 06:00:00	2.273 GB
2020-08-12 08:00:00	2.277 GB
2020-08-12 10:00:00	2.281 GB
2020-08-12 12:00:00	2.286 GB
2020-08-12 14:00:00	2.290 GB
2020-08-12 16:00:00	2.294 GB
2020-08-12 18:00:00	2.299 GB
2020-08-12 20:00:00	2.303 GB
2020-08-12 22:00:00	2.308 GB
2020-08-13 00:00:00	2.312 GB
2020-08-13 02:00:00	2.316 GB
2020-08-13 04:00:00	2.320 GB
2020-08-13 06:00:00	2.324 GB
2020-08-13 08:00:00	2.328 GB
2020-08-13 10:00:00	2.332 GB
2020-08-13 12:00:00	2.337 GB
2020-08-13 14:00:00	2.342 GB
2020-08-13 16:00:00	2.346 GB
2020-08-13 18:00:00	2.351 GB
2020-08-13 20:00:00	2.355 GB
2020-08-13 22:00:00	2.359 GB
2020-08-14 00:00:00	2.363 GB
2020-08-14 02:00:00	2.368 GB
2020-08-14 04:00:00	2.372 GB
2020-08-14 06:00:00	2.376 GB
2020-08-14 08:00:00	2.380 GB

- NSD SIZE Values

Time	mean
2020-08-12 02:00:00	2.621 GB
2020-08-12 04:00:00	2.625 GB
2020-08-12 06:00:00	2.629 GB
2020-08-12 08:00:00	2.633 GB
2020-08-12 10:00:00	2.637 GB
2020-08-12 12:00:00	2.642 GB
2020-08-12 14:00:00	2.646 GB
2020-08-12 16:00:00	2.650 GB
2020-08-12 18:00:00	2.655 GB
2020-08-12 20:00:00	2.659 GB
2020-08-12 22:00:00	2.664 GB
2020-08-13 00:00:00	2.668 GB
2020-08-13 02:00:00	2.671 GB
2020-08-13 04:00:00	2.676 GB
2020-08-13 06:00:00	2.680 GB
2020-08-13 08:00:00	2.684 GB
2020-08-13 10:00:00	2.688 GB
2020-08-13 12:00:00	2.693 GB
2020-08-13 14:00:00	2.698 GB
2020-08-13 16:00:00	2.702 GB
2020-08-13 18:00:00	2.706 GB
2020-08-13 20:00:00	2.711 GB
2020-08-13 22:00:00	2.715 GB
2020-08-14 00:00:00	2.719 GB
2020-08-14 02:00:00	2.723 GB
2020-08-14 04:00:00	2.728 GB
2020-08-14 06:00:00	2.732 GB
2020-08-14 08:00:00	2.736 GB

EXPERIMENTO IV - MEMORY LEAKS

devices

Fabric Reachability Status

No Fabric degradation detected now

FPC Type	Configured Deg	Current Deg	Degradation/slot	DegPlanes for Line Rate
FPC_0 : MPC3E NG PQ & Flex Q	MPC3E NG PQ & Flex Q	n/a,none	none	4/6
FPC_1 : MPC3E NG PQ & Flex Q	MPC3E NG PQ & Flex Q	n/a,none	none	4/6
FPC_10 : MPC3E NG PQ & Flex Q	MPC3E NG PQ & Flex Q	n/a,none	none	4/6
FPC_11 : MPC3E NG PQ & Flex Q	MPC3E NG PQ & Flex Q 2020-11-13 17:00:00 to 2020-11-13 17:30:00	n/a,none	none	4/6
FPC_2 : MPC3E NG PQ & Flex Q	MPC3E NG PQ & Flex Q	n/a,none	none	4/6
FPC_3 : MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE	MPC7E 3D 40XGE	n/a,none	none	6/6
FPC_4 : MPC7E 3D 40XGE	MPC7E 3D 40XGE	n/a,none	none	6/6
FPC_5 : MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE		n/a,none	none	6/6
FPC_7 : MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE		n/a,none	none	6/6
FPC_8 : MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE		n/a,none	none	6/6
FPC_9 : MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE		n/a,none	none	6/6

Fabric Degradation State

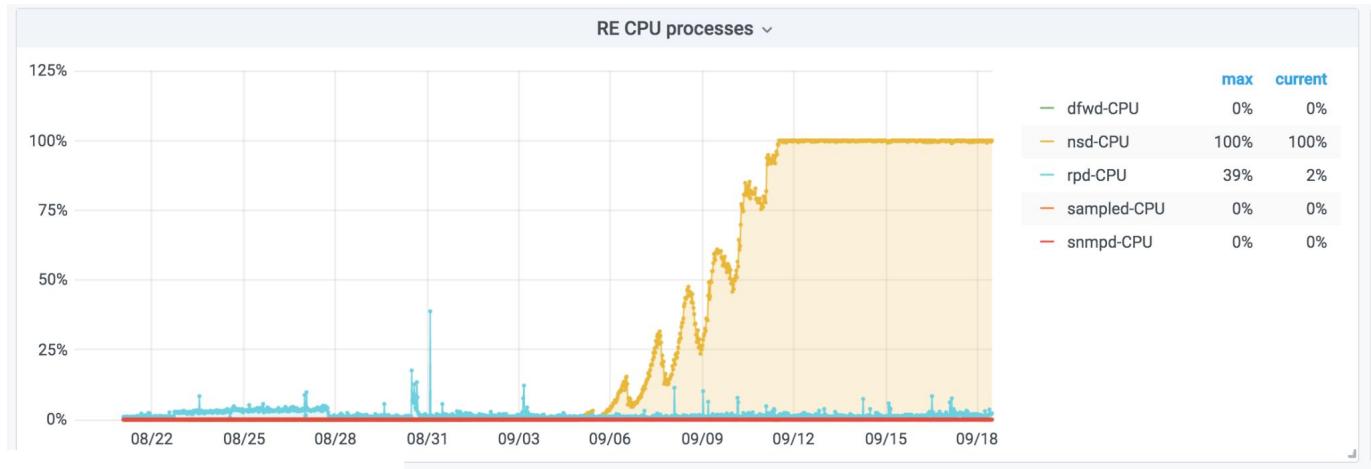
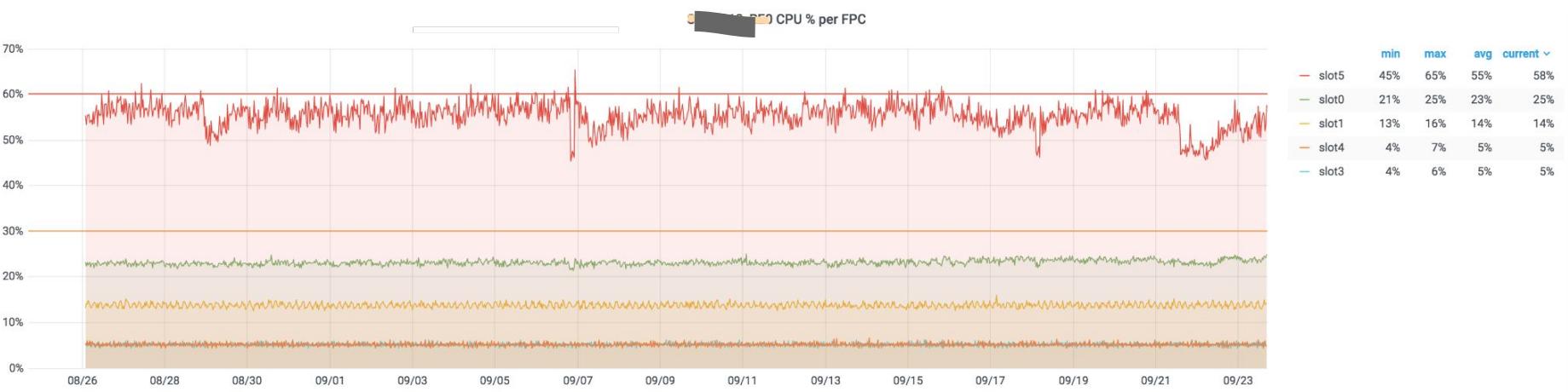
slot 0	Online
slot 1	Online
slot 10	Online
slot 11	Online
slot 2	Online
slot 3	Online
slot 4	Online
slot 5	Online
slot 6	--
slot 7	Online
slot 8	Online
slot 9	Online

11/12 01:00 11/12 13:00 11/13 01:00 11/13 13:00

Fabric Degradation State	Configured Deg	Current Deg	Degradation/slot	DegPlanes for L...
MPC3E NG PQ & Flex Q	slot0 – n/a,none	slot0 – none	slot0 – none	slot0 – 4/6
MPC7E 3D MRATE-12xQSFP-XGE-XLGE-CGE	slot4 – n/a,none	slot4 – none	slot4 – none	slot4 – 6/6
MPC7E 3D 40XGE	slot5 – n/a,none	slot5 – none	slot5 – none	slot5 – 6/6

FPC 5 -----

EXPERIMENTO V - MONITORIZACION EXTREMA



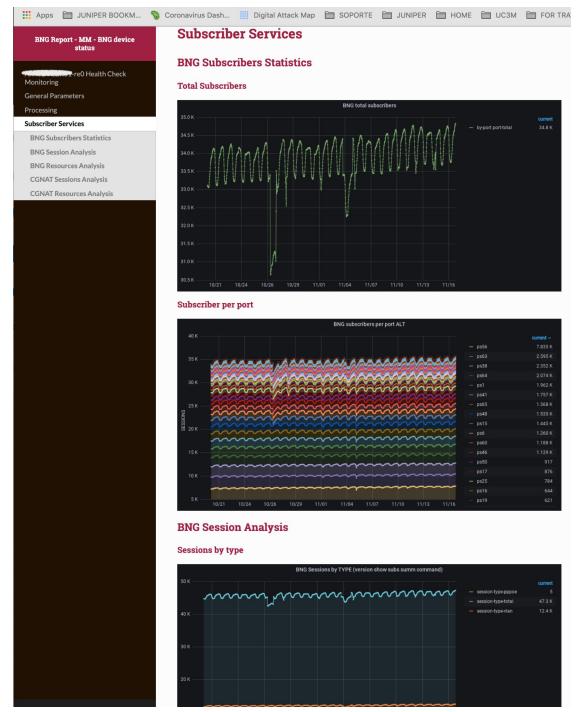
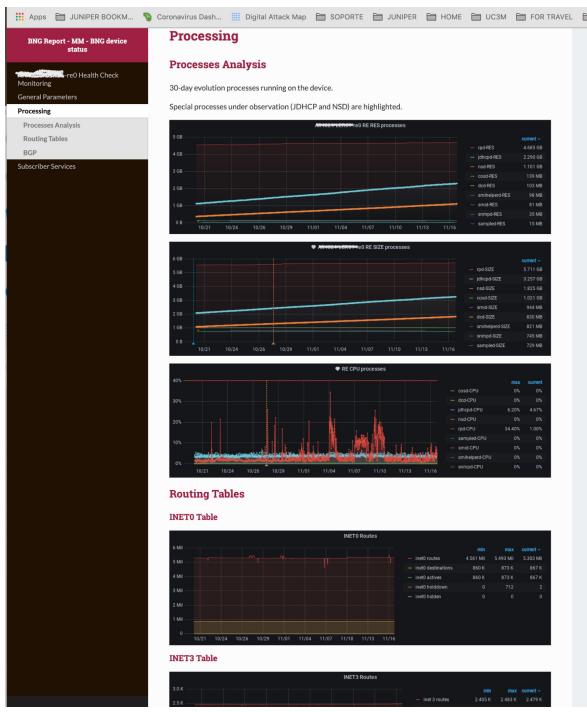
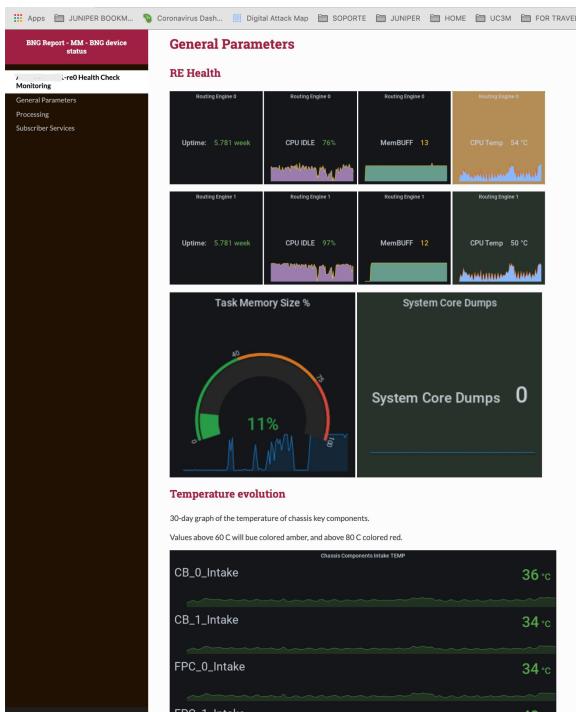
EXPERIMENTO V - MONITORIZACION EXTREMA

Para una segunda parte (en preparación, ahora mismo)

Integra gráficas de grafana, R y consultas a la db de Influxdb.

EXPERIMENTO VI - REPORTING

Ejemplo



EXPERIMENTO VI - REPORTING

CONCLUSIÓN

open-nti-new puede ser una herramienta útil para:

- comprobaciones y chequeos de equipos específicos, o subnets/funcionalidades
- automatización de tareas de Health-check en redes
- ensayos de visualización de mediciones de red

necesita dedicación para ser operativa

necesita añadidos (para hacer reporting, por ejemplo)

GRACIAS!!