WORKSHOP GARR 2022

NET MAKERS

Telemetria, logging e alerting in GARR-T

Giovanni Cesaroni, Gianni Marzulli

GARR

Agenda

01 Motivazioni, obiettivo e percorso

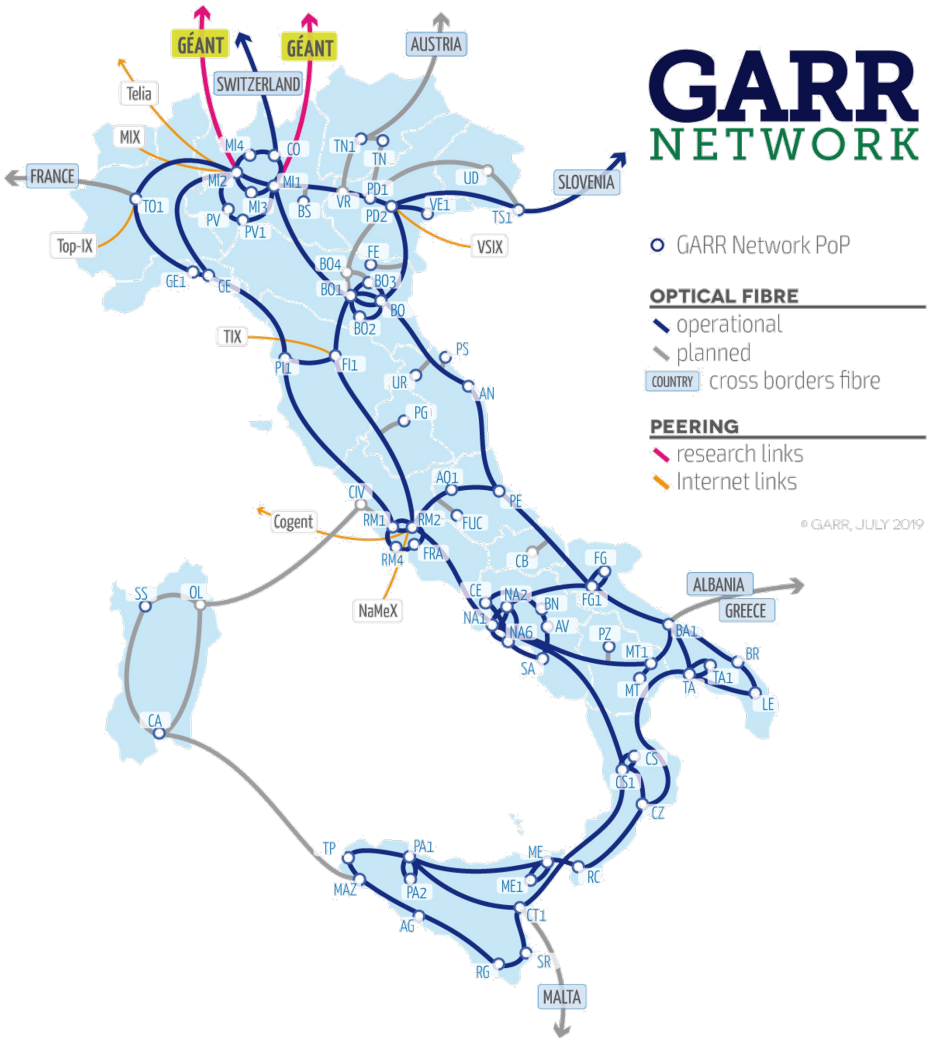
02 Architettura, implementazione e deployment

03 Alerting e downsampling

04 Use case: packet network

05 Direzione futura

GARR-T: una rete da osservare



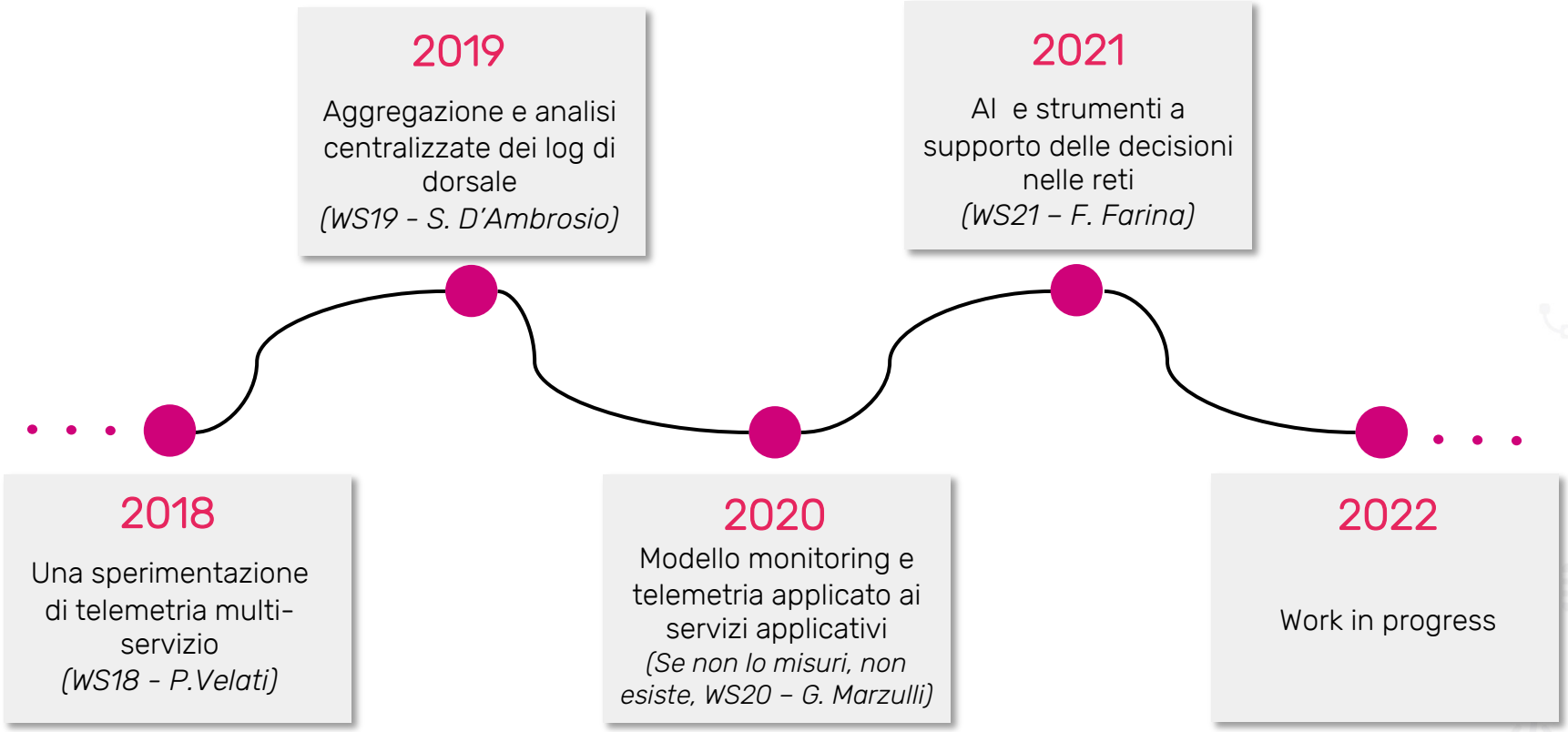
La rete:

- 20,000 km di fibra
- 100 PoP, 10 DC
- 100 apparati ottici, 130 a pacchetto

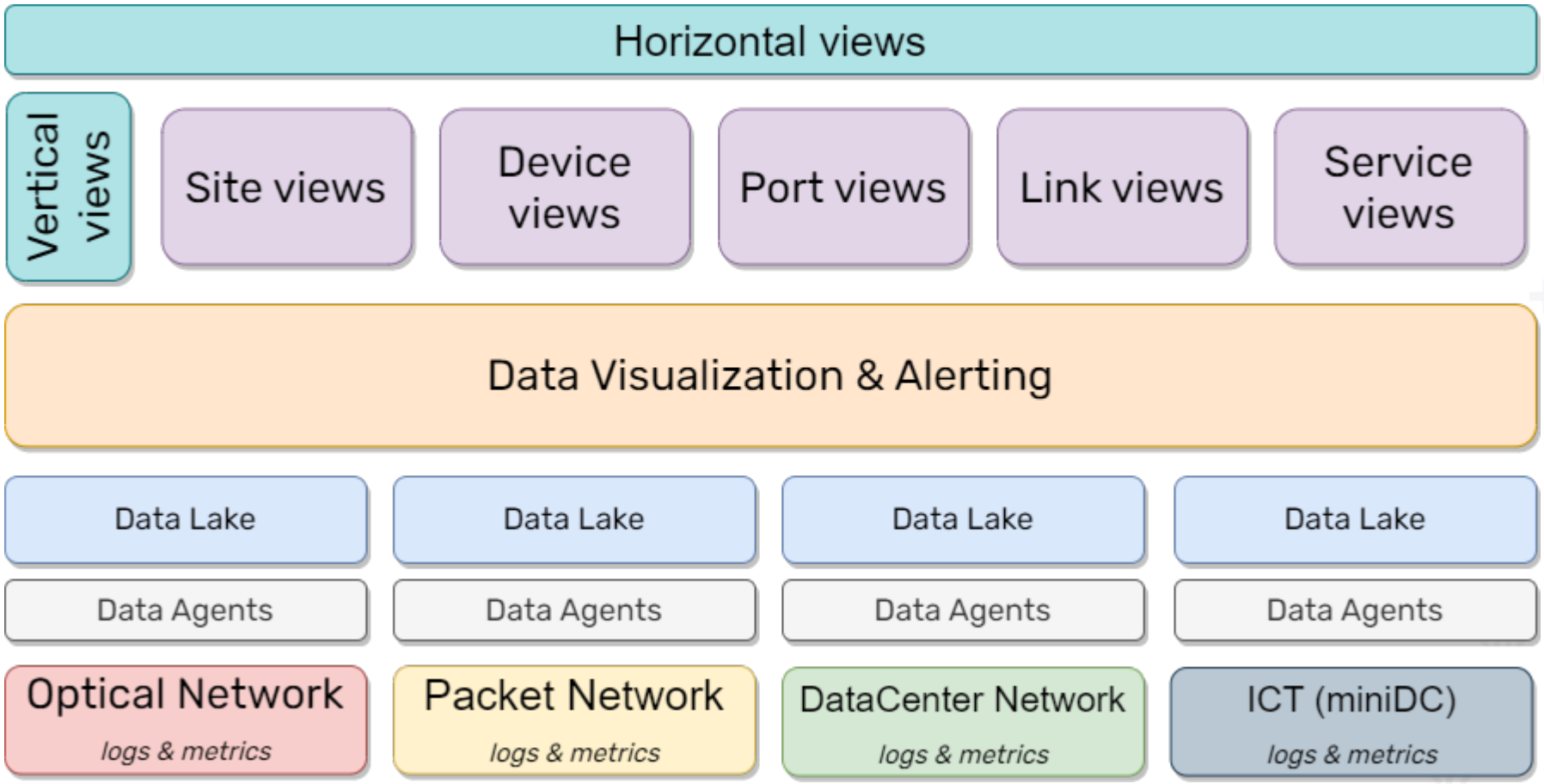
Obiettivo:
Modello e strumenti di osservabilità per monitoraggio e automazione:

- Rete ottica, rete pacchetto, rete DataCenter, ICT
- Base costitutiva della overlay GARR-T

Un processo iniziato qualche anno fa

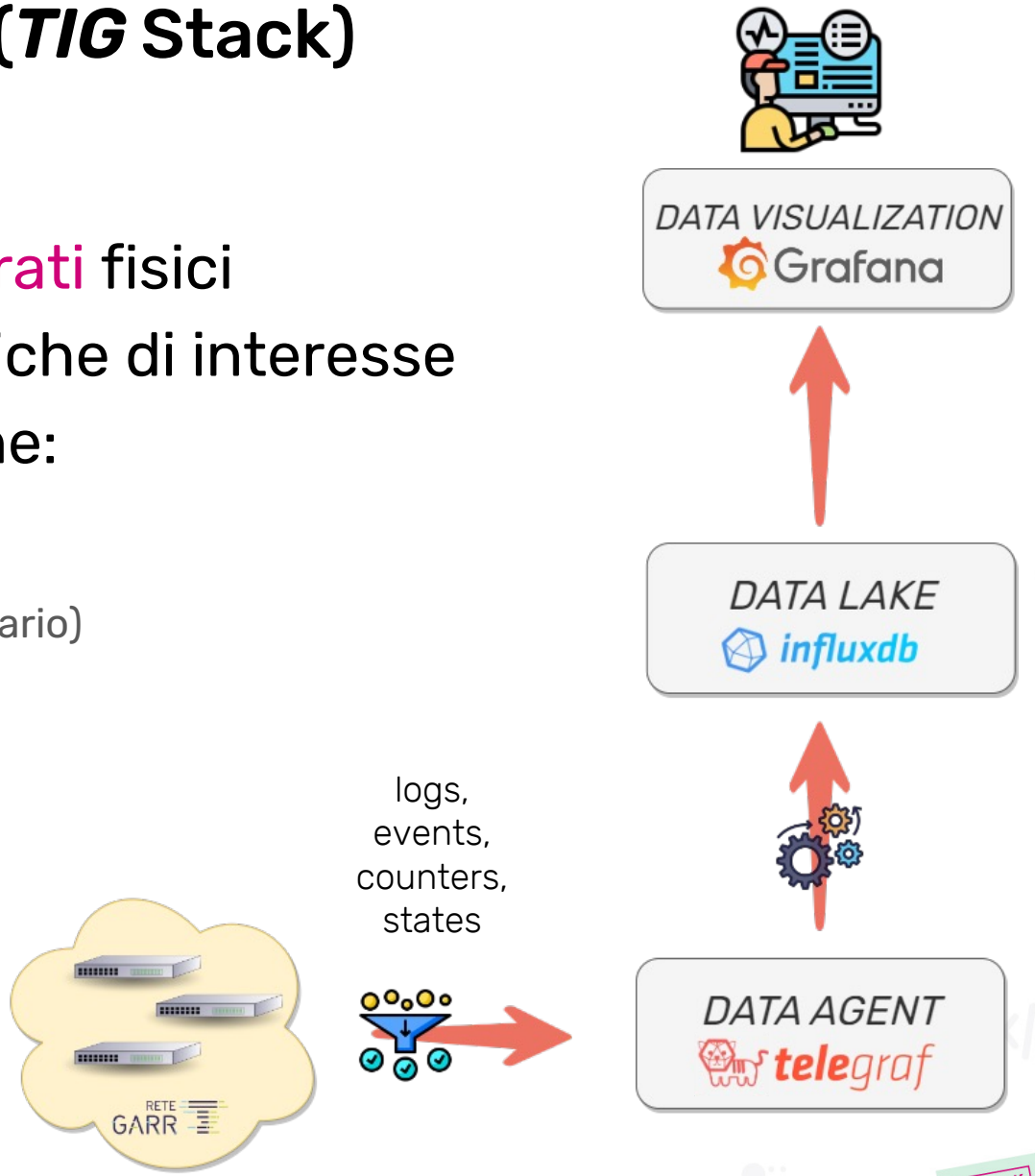


Modello architetturale

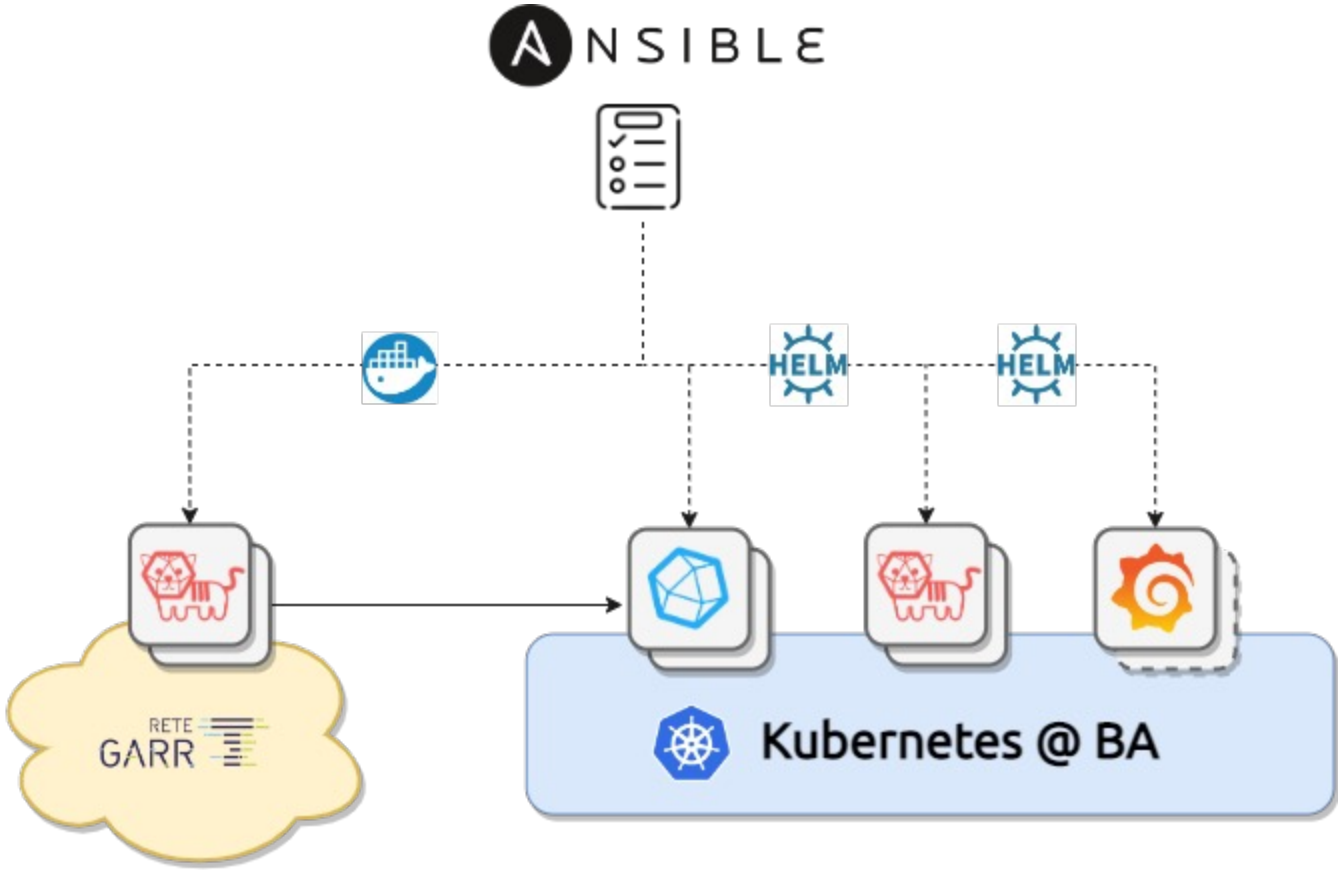


Implementazione (*TIG* Stack)

- Sorgente dati: apparati fisici
- Scouting delle metriche di interesse
- Canali di acquisizione:
  - gNMI
  - SNMP (ancora necessario)
  - SYSLOG

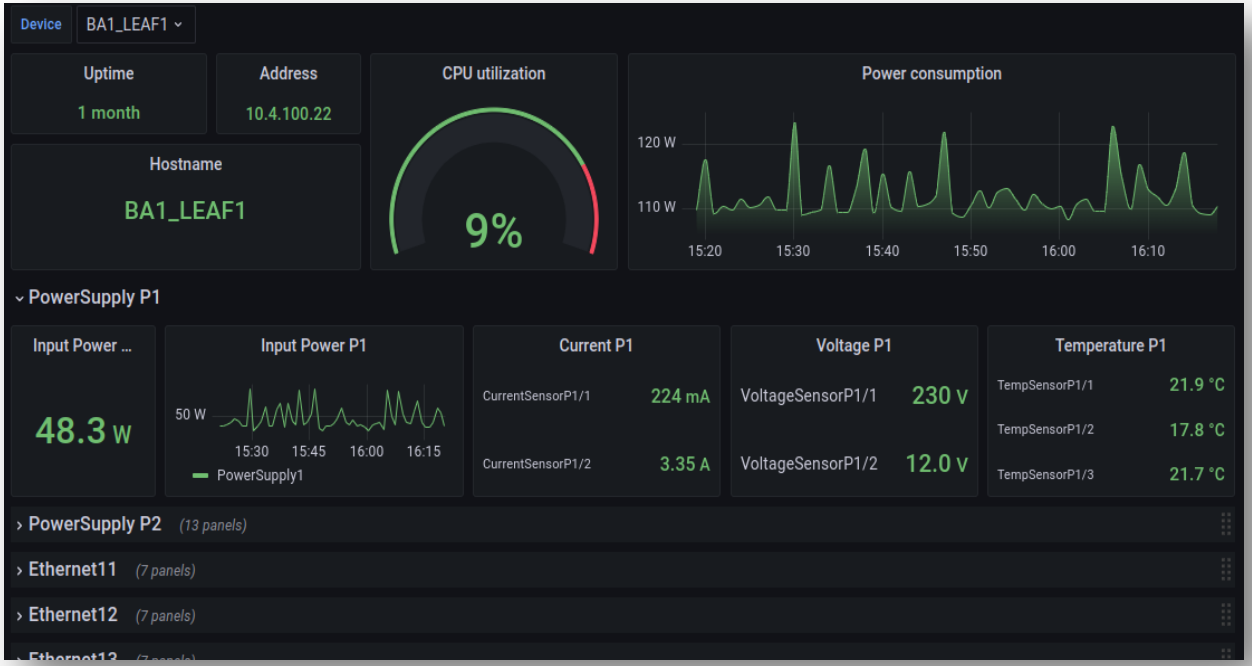


Automated deployment



Cosa osserviamo: Stato apparato

- Utilizzo CPU, Temperature, assorbimento energetico, stato componenti

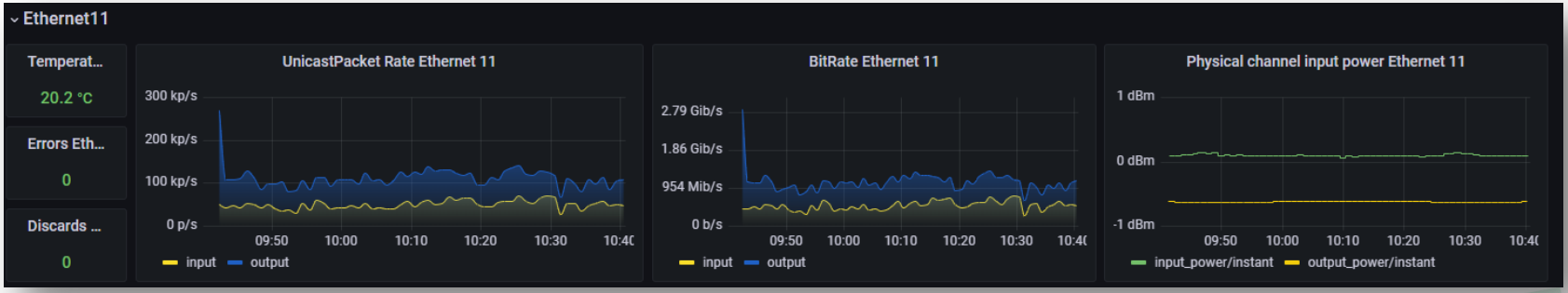


DataCenter net: Device view

Cosa osserviamo: Stato porte e Traffico I/O

- Stati e contatori porte: admin/oper status, temperature, ottetti e pacchetti, parametri ottici, errori

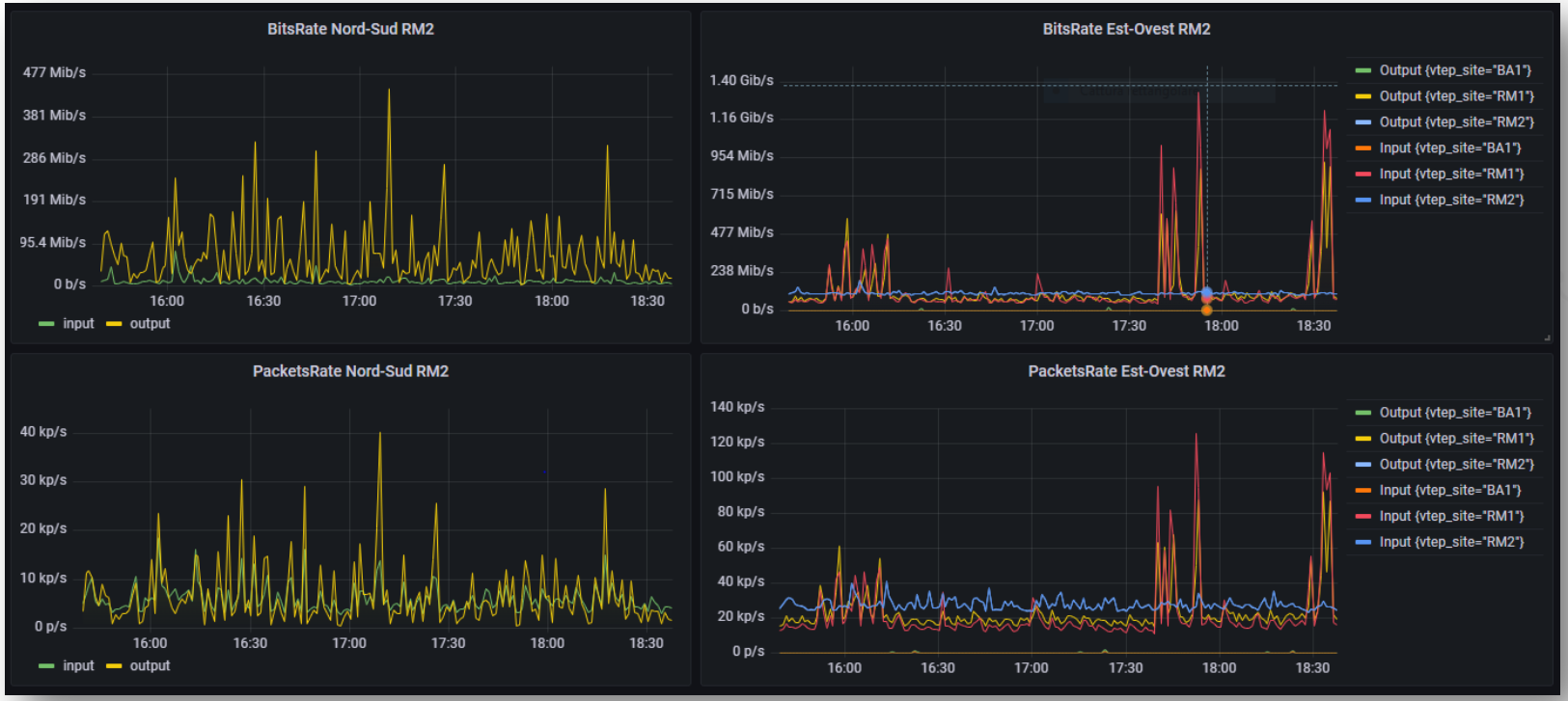
Optical net: port view



DataCenter net: port view

Cosa osserviamo: Traffico I/O aggregato

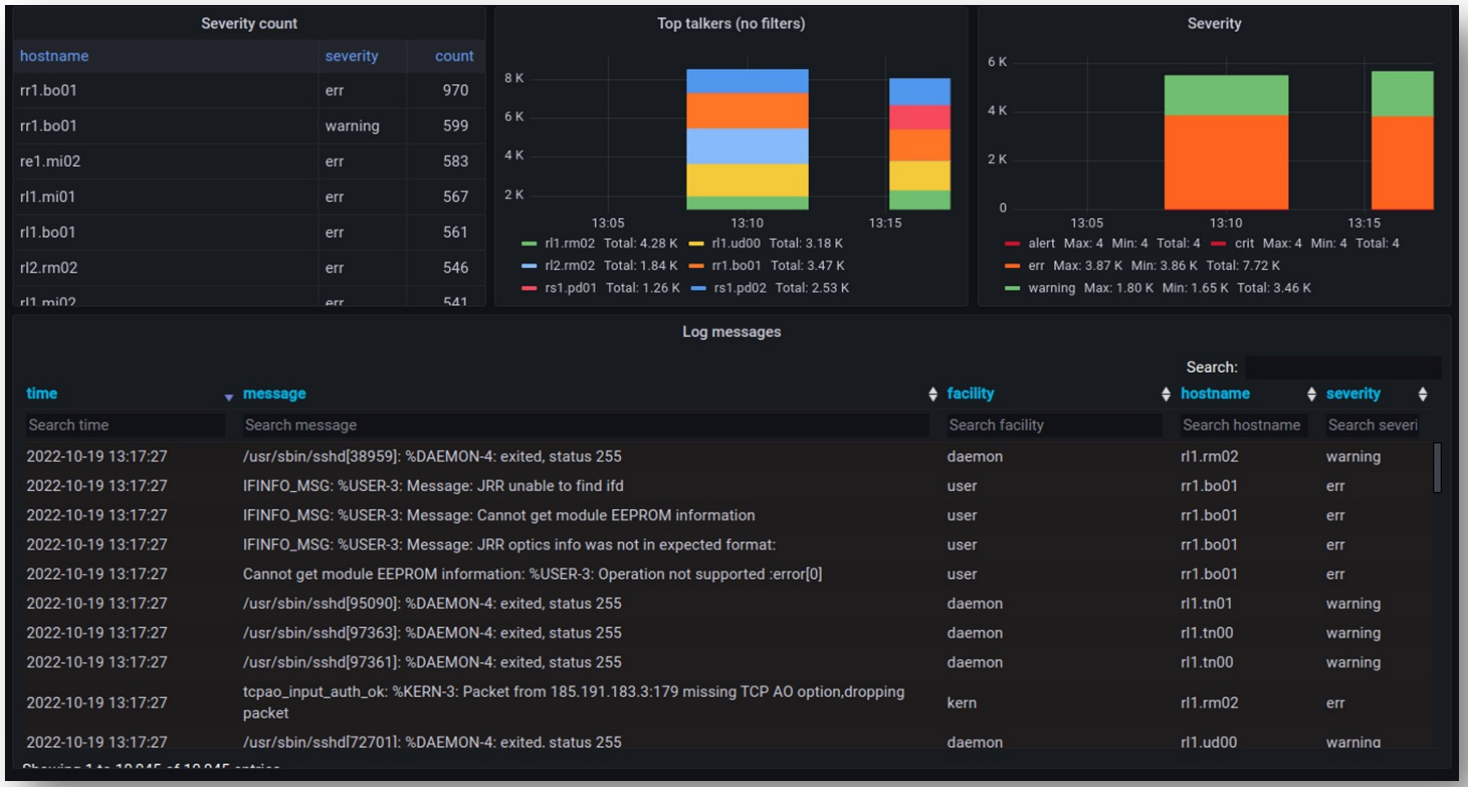
- Traffico Nord-Sud, Est-Ovest



DataCenter net: site view

Cosa osserviamo: Log ed eventi

- Facility: kernel, firewall, daemon, security, user
  - Top talkers, monitoraggio per serverity

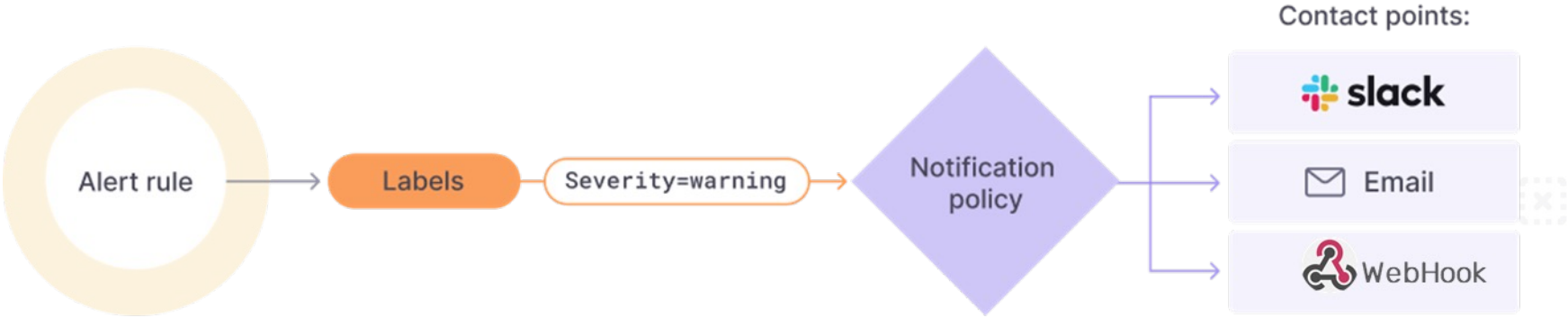


Log backbone

Alerting & Notifying

Gestione alerting centralizzata in Grafana

- Layer comune a tutte le sorgenti
- Alert manager migliorato in versione 9
- Alert history



Alerting & Notifying

Rilevazione anomalie ed eventi di interesse tramite regole:

if *condizione* *then* *azione*

METRIC > *TRESHOLD* for *INTERVAL*

```
from(bucket: "optical_telemetria")
|> range(start: -2m, stop: now())
|> filter(fn: (r) => r["_measurement"] == "OTN")
|> filter(fn: (r) =>
r["_field"] == "och-bit-error-rate-pre-fec")
|> last()
```

> 0.2 for 10s

Fire *ALERT*

notification

webhook

Data retention and downsampling

Short-term

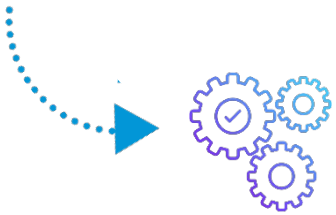
Dettagli utili solo nel breve periodo

- Campionamento real-time: 2s – 1m
- Data retention: 30d

Long-term

Archiviazione per fini statistici

- Selezione metriche: traffic I/O, error rate, parametri ottici
- Aggregazione: max, avg e min ogni 15m - 3h
- Data retention: infinita



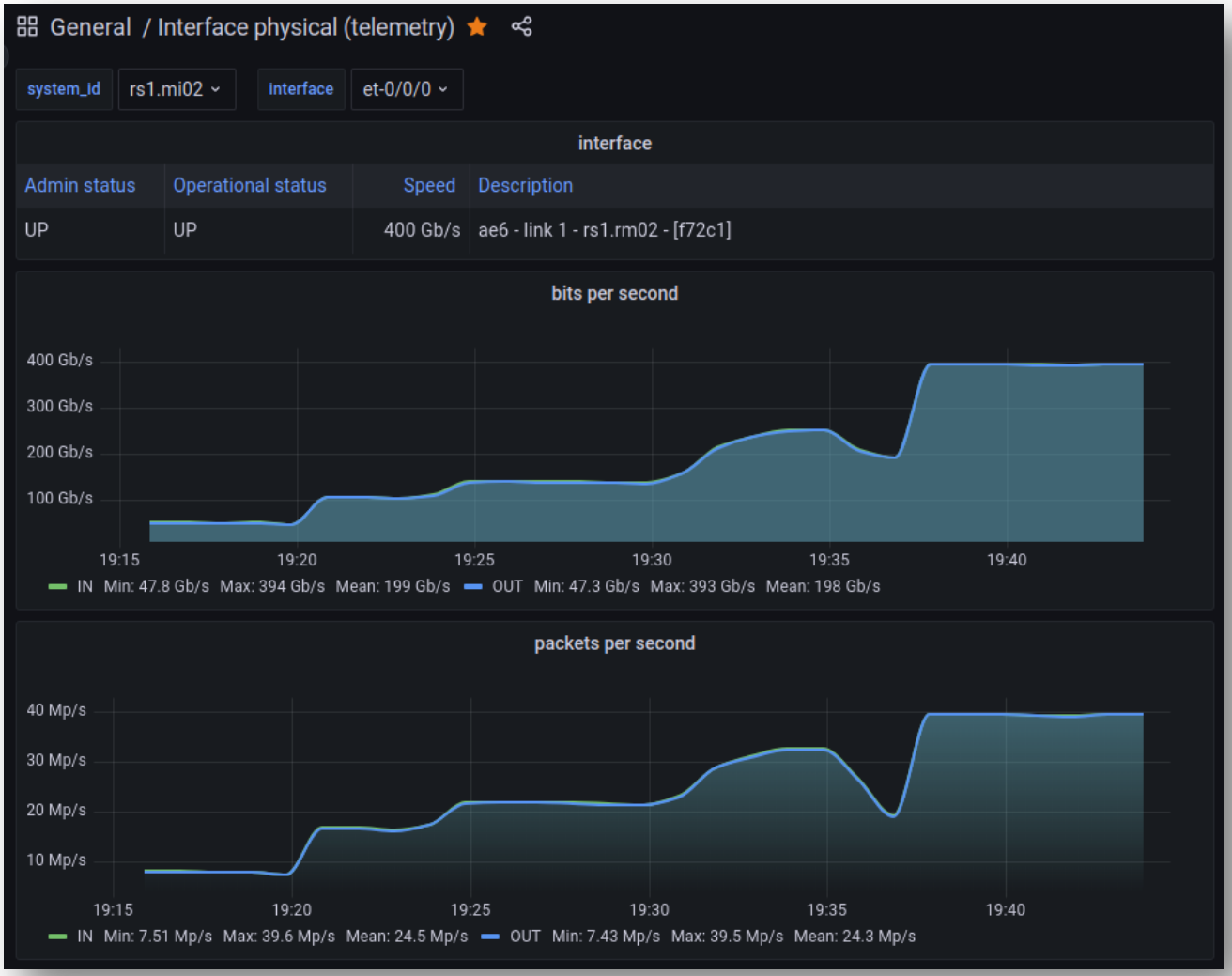
Downsampling tramite task Flux in bucket dedicati

Packet Network

Use case: packet network

- Applicare il modello descritto al monitoring della rete a pacchetto
- Eliminare limiti di scalabilità nell'acquisizione delle metriche
- Dotare NOC/OPS di strumenti "real time" per la migrazione GARR-T

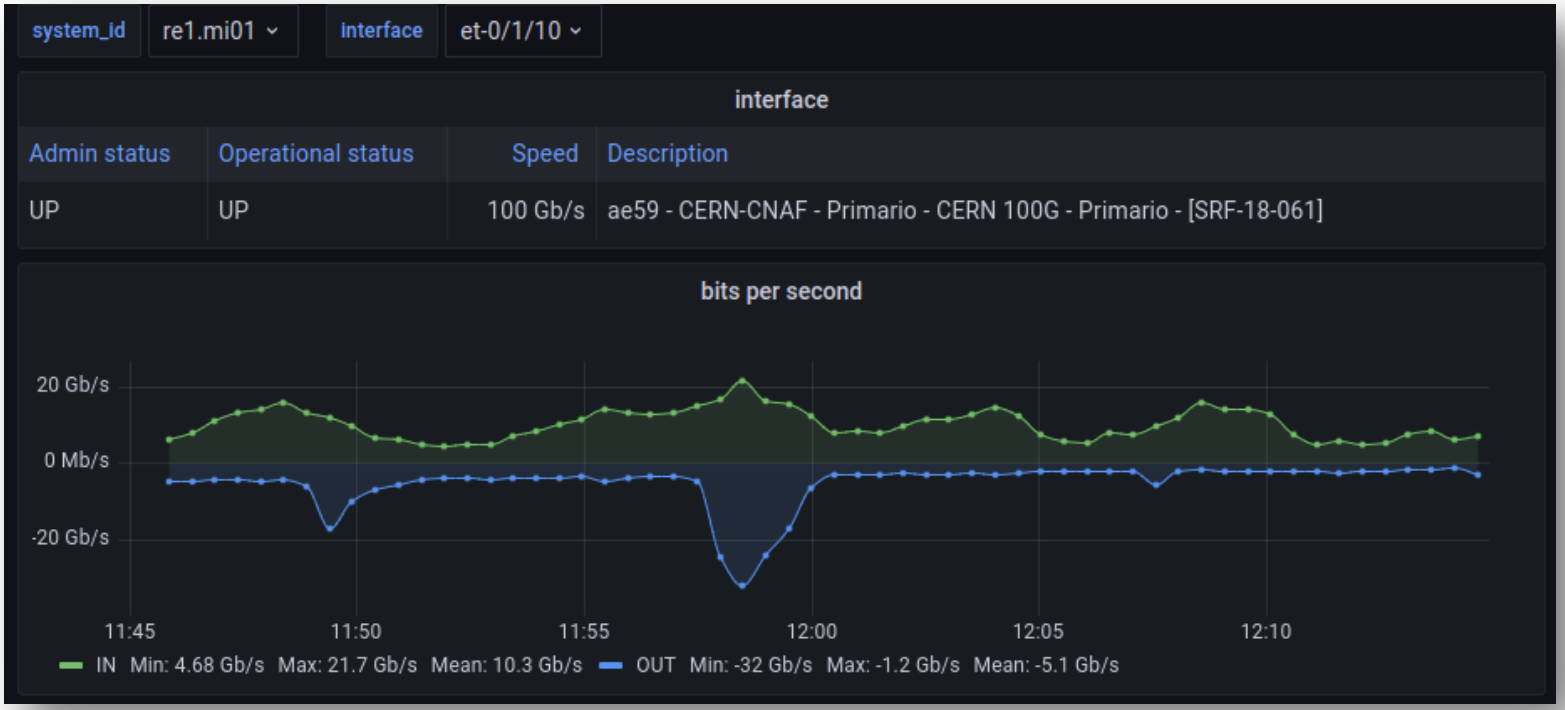
GARR-T primo test di carico (30/09/2022)



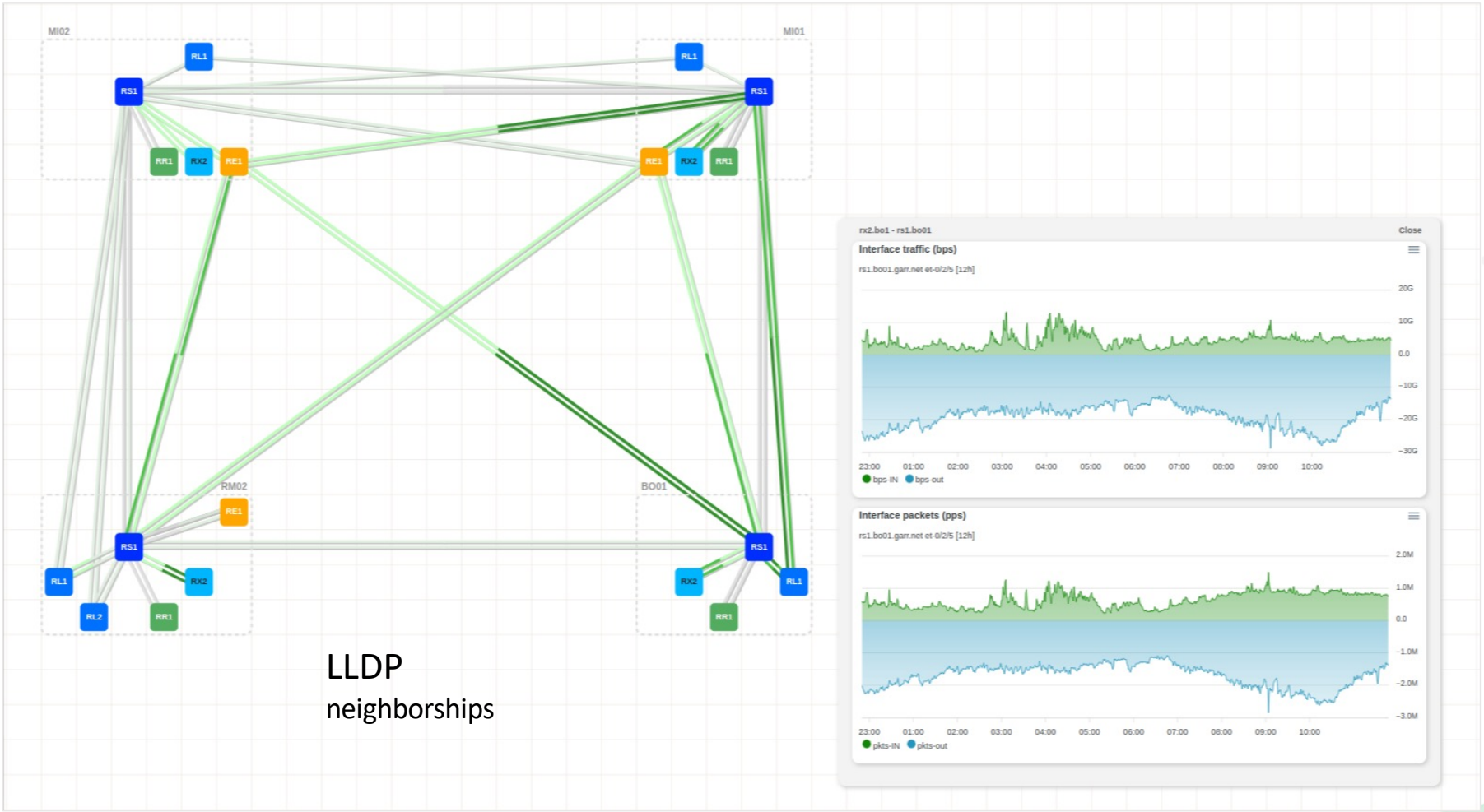
Da SNMP a GNMI

SNMP (5') **vs** **GNMI (1')**

30”


system_id
re1.mi01
interface
et-0/1/10
interface
Admin status
Operational status
Speed
Description
UP
UP
100 Gb/s
ae59 - CERN-CNAF - Primario - CERN 100G - Primario - [SRF-18-061]
bits per second
20 Gb/s
0 Mb/s
-20 Gb/s
11:45
11:50
11:55
12:00
12:05
12:10
IN Min: 4.68 Gb/s Max: 21.7 Gb/s Mean: 10.3 Gb/s
OUT Min: -32 Gb/s Max: -1.2 Gb/s Mean: -5.1 Gb/s

Visione real time della topologia e stato della rete



Elementi abilitanti

- Acquisizione dati:
  - Telemetria: GNMI
  - Telegraf: JTI input plugin
- Scrittura e lettura dati:
  - InfluxDB
  - Flux data language
- Rappresentazione dati:
  - Grafana
  - Gins

GNMI - gRPC - YANG models

telegraf

influxdb

FLUX |>

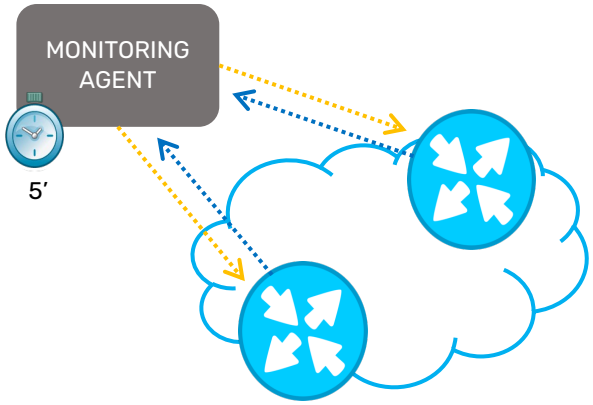
Grafana

GINS

Telemetria da SNMP a GNMI

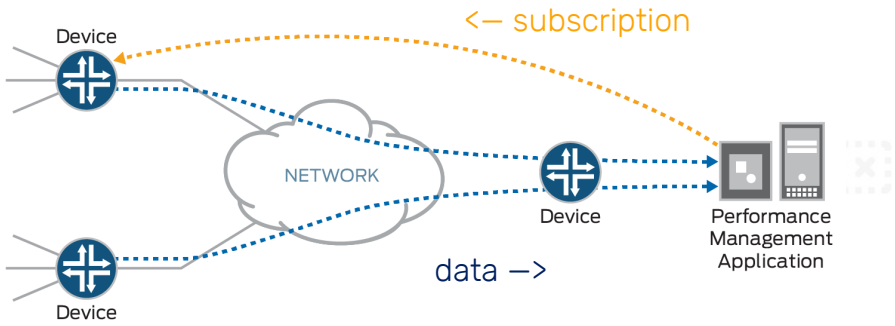
Simple Network Management Protocol

- pull
- synchronous
- slow rates
- born in 1988
- data model: MIBs

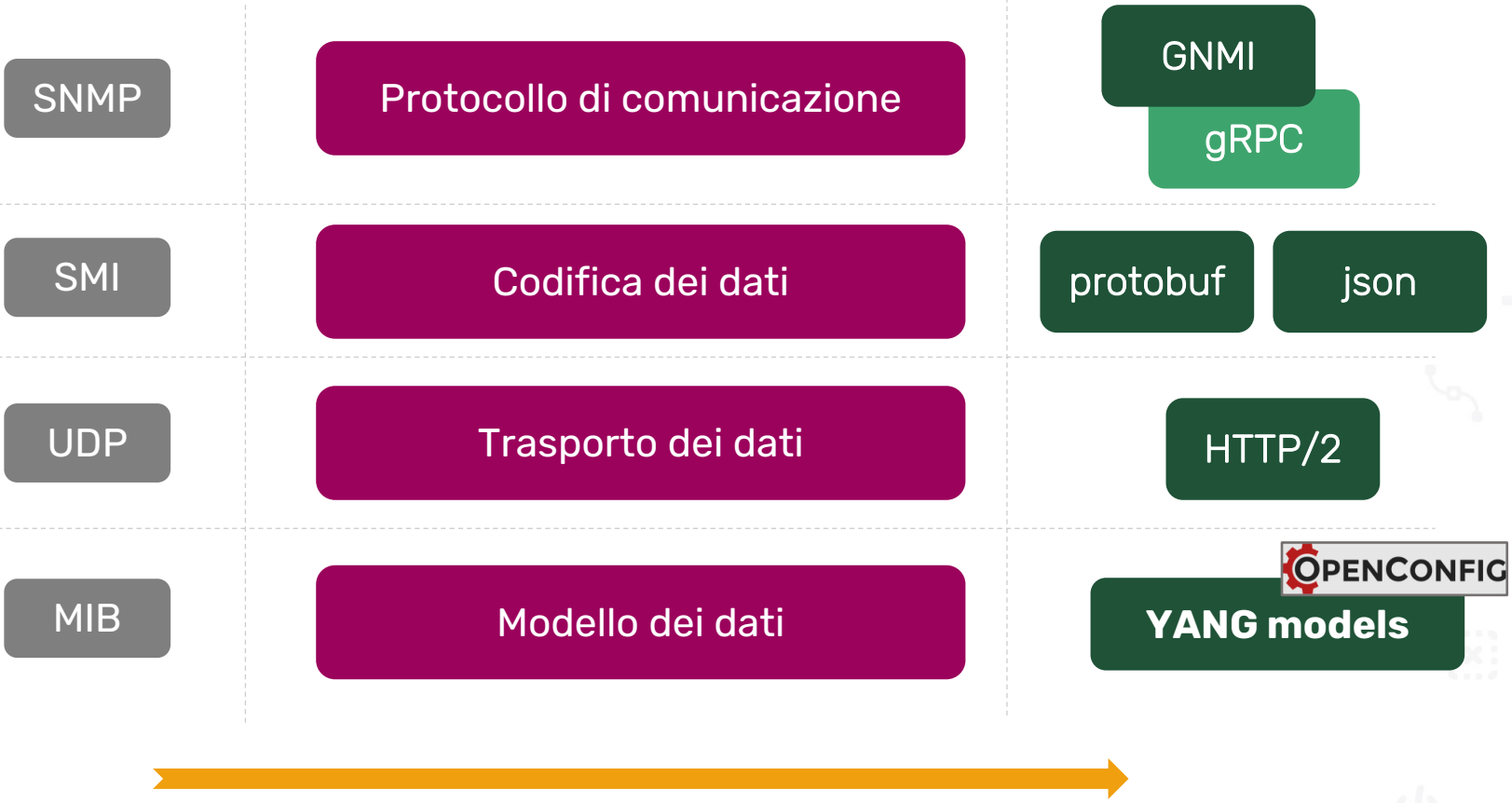


gRPC Network Management Interface

- push
- asynchronous
- fast rates
- born in 2015
- data model: YANG

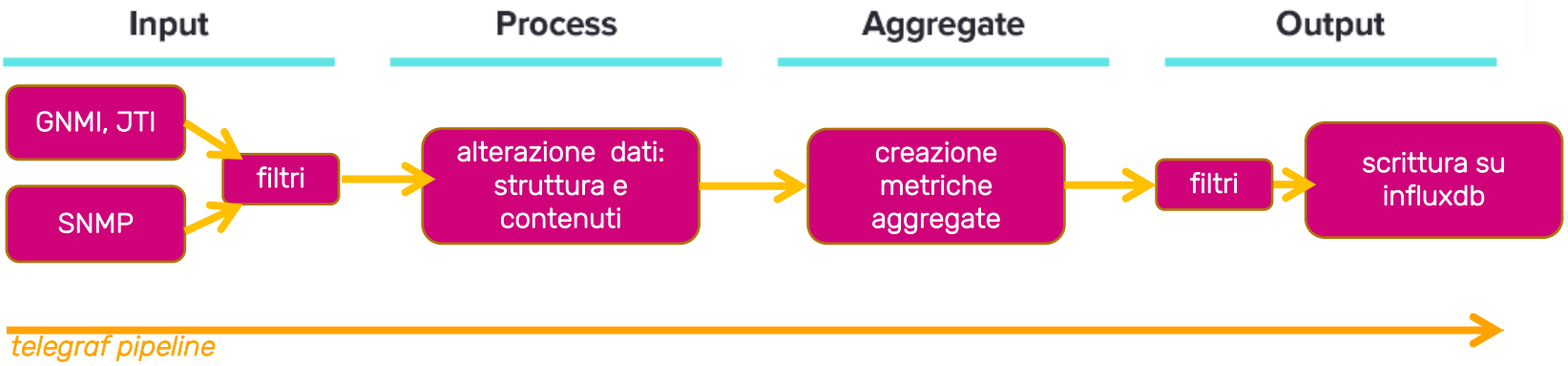


Telemetria con GNMI

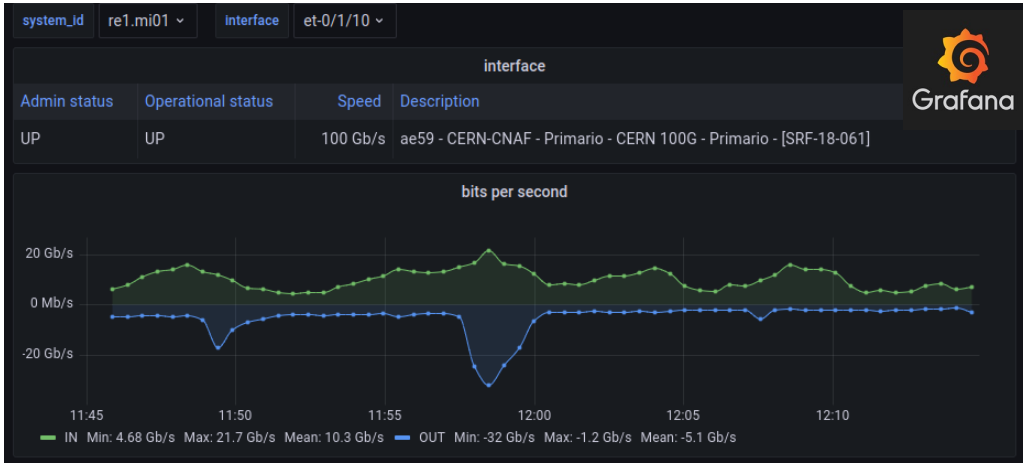


Telegraf pipeline

telegraf™



Flux data language



SOURCES

```
from(bucket: "telegraf-interface")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r._measurement == "/interfaces/")
  |> filter(fn: (r) =>
                r._field == "/interfaces/interface/state/counters/in-octets" or
                r._field == "/interfaces/interface/state/counters/out-octets")
  |> filter(fn: (r) => r.system_id == "${system_id}")
  |> filter(fn: (r) => r.interface == "${interface}")
  |> derivative(unit: 1s, nonNegative: true)
  |> pivot(rowKey: ["_time"], columnKey: ["_field"], valueColumn: "_value")
  |> map(fn: (r) => ({ r with loadIn: r["/interfaces/interface/state/counters/in-octets"] * 8.0 / 1000000.0 }))
  |> map(fn: (r) => ({ r with loadOut: r["/interfaces/interface/state/counters/out-octets"] * (-8.0) / 1000000.0 }))
  |> keep(columns:["_time","loadIn","loadOut"])
```

filters process

flux pipeline

Dove stiamo andando

- Strumenti sempre piu efficienti per la gestione rete
- Modello uniformemente applicato e a tutti i livelli della rete
- Correlazione dell'informazione proveniente dai vari layer
- Osservabilità al servizio dell'utente e della programmabilità

GRAZIE

Credits:

- https://unsplash.com
- https://flaticon.com