Infrastruttura BDP del CNAF

Enrico Fattibene - Antonio Falabella - Lucia Morganti

29 marzo 2023

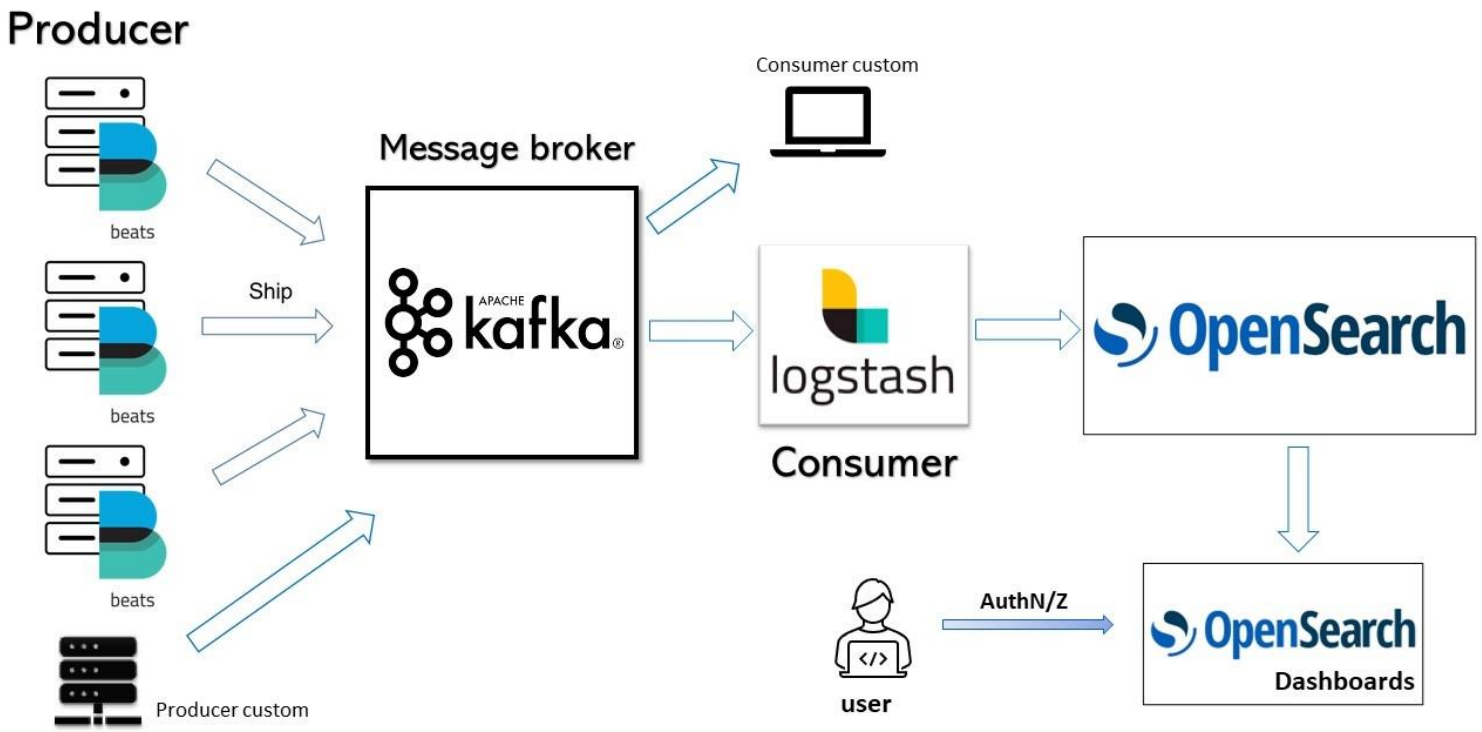
Sommario

- Motivazioni e scopo del lavoro
- Architettura e strumenti utilizzati
- Configurazione servizi
- Integrazione con servizi esistenti
- Possibili utilizzi della piattaforma
- Demo

Motivazioni e scopo

- Infrastruttura **modulare** di gestione e analisi dati eterogenei a servizio di amministratori ed utenti del CNAF
- A partire dal lavoro già fatto al CNAF su Big Data Platform
- Lavoro in collaborazione con diversi reparti del CNAF
  - A. Falabella, E. Fattibene, F. Fornari - Storage
  - S. Antonelli - SSNN
  - D. Michelotto - Farming
  - D. Lattanzio - User Support
  - V. Ciaschini, F. Amori - Security
- Attività nell'ambito del progetto CNAF Reloaded
  - Task Operations
  - Infrastruttura di base **general purpose** + caso d'uso analisi dei log

Schema semplificato

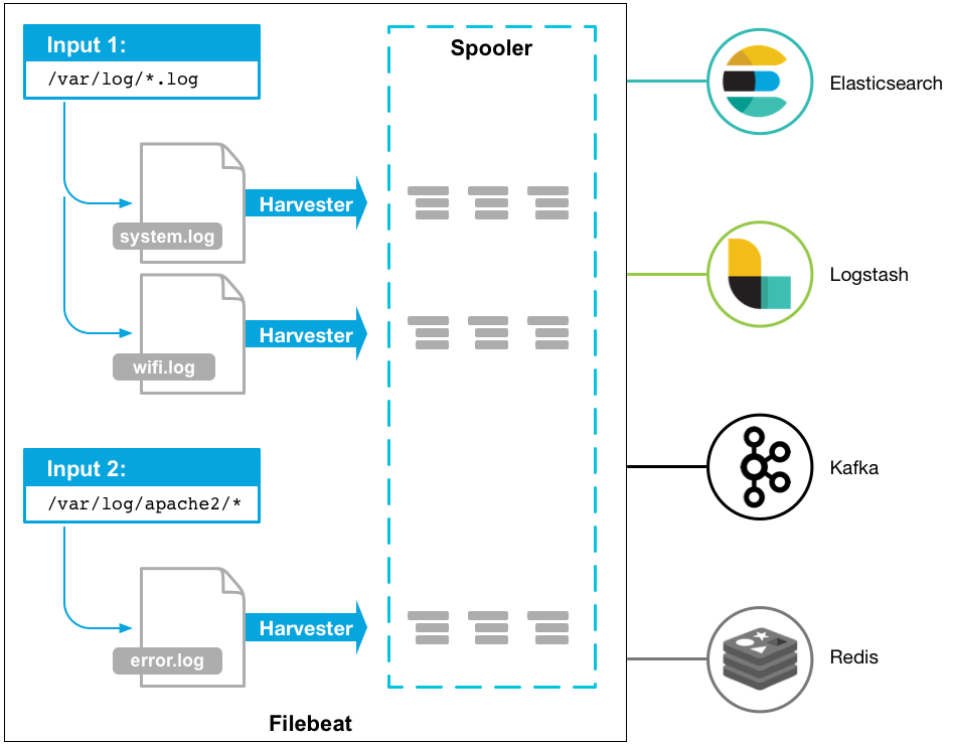


Hardware

- Tre nodi virtuali per **Kafka** (load balancing + HA) su infrastrutture separate
  - cnlog-kafka01, cnlog-kafka02, cnlog-kafka03 (1.8TB spazio totale)
- Tre nodi virtuali per la gestione del **cluster OpenSearch**
  - cnlog-master01, cnlog-master02, cnlog-master03
- Quattro nodi fisici per i **dati di OpenSearch**
  - cnlog-data[01-04] (8TB spazio totale su SSD) → facilmente scalabile
- Un nodo virtuale **Dashboard**
  - cnlog-dashboards01
- Un nodo virtuale **Logstash**
  - cnlog-logstash01 → facilmente scalabile

Filebeat

- Producer specifico per spedire **log** (log shipper), nel nostro caso verso Kafka
- Configurazione semplice **via Puppet**
  - Output verso Kafka criptato via SSL
  - Autenticazione su Kafka via username e password
  - Ogni reparto/progetto scrive solo su topic concordato con amministratori
  - **Tag** usato per indicizzare su OpenSearch



Custom producer

- E’ possibile utilizzare **altri tipi di producer** di dati
  - Tool che supportano Kafka versione 3.3.1
  - Software sviluppato ad hoc usando librerie che parlano con Kafka (ad es. python)

```
from kafka import KafkaProducer
import json
import time
from bson import json_util
import random
import time

producer = KafkaProducer(
     bootstrap_servers=['cnlog-
kafka01.cr.cnaf.infn.it:9192'],
     security_protocol="SASL_SSL",
     sasl_plain_username='test',
     sasl_plain_password='test',
     sasl_mechanism='PLAIN',
     ssl_cafile='./ca.pem',
)
```

```
topicName = 'testTopic'

while True:
  ts = time.time()
  data = { 'tag ': 'blah',
        'name' : 'python_consumer',
        'index' : 1,
        'number' : random.randint(0,9),
        'timestamp' : ts
        }
  ack = producer.send(topicName, json.dumps(data,
default=json_util.default).encode('utf-8'))
  metadata = ack.get()
  print(data['number'])
  print(data['timestamp'])
  print(metadata)
  time.sleep(3)
```

Kafka - overview

- Piattaforma open source per lo **streaming di eventi**
- Può essere usato come servizio a se
- Configurato in HA con tre nodi
  - Tre VM su tre infrastrutture diverse (storage, farming oVirt, farming VMWare)
  - Spazio disco di 1.8 TB
  - Scalabile orizzontalmente aggiungendo altri nodi in caso di bisogno
- **Topic** diversi per reparti/progetti diversi
  - Suddivisione logica dei dati
- Numero **repliche**, **partition** e **retention** configurabili
  - Di default replica 3, partition 3 e retention di una settimana
- Dati possono essere letti da diversi **consumer**
  - Eventualmente organizzati in consumer groups

Kafka - autorizzazione

- L’autorizzazione è gestita tramite **ACL**
  - Di default solo admin accede a tutti i topic
  - ACL sui topic

```
Current ACLs for resource `ResourcePattern(resourceType=TOPIC, name=servnaz,
patternType=LITERAL)`:
        (principal=User:servnaz, host=*, operation=DESCRIBE, permissionType=ALLOW)
        (principal=User:servnaz, host=*, operation=WRITE, permissionType=ALLOW)
        (principal=User:logstash, host=*, operation=READ, permissionType=ALLOW)
        (principal=User:logstash, host=*, operation=DESCRIBE, permissionType=ALLOW)
        (principal=User:servnaz, host=*, operation=READ, permissionType=ALLOW)
```

  - ACL sui consumer groups

```
Current ACLs for resource `ResourcePattern(resourceType=GROUP, name=logstash,
patternType=LITERAL)`:
         (principal=User:logstash, host=*, operation=READ,
    permissionType=ALLOW)
```

Logstash

- Al momento unica istanza configurata per leggere da diversi topic di Kafka
  - A regime: un servizio Logstash per reparto/progetto
- Configurata in HA (tre nodi virtuali)
- **Filtraggio/arricchimento dei dati**
  - Estrazione info rilevanti che diventano campi degli indici OpenSearch
  - Costruzione nuove info in seguito ad operazioni su info esistenti
  - Rimozione record non rilevanti
  - Sincronizzazione orario entry nell'indice con orario del log
  - Possibile invio dei record arricchiti su nuovo topic Kafka
- Invio ad OpenSearch via SSL
  - Autenticazione con user e password

Custom consumer

- Come per i producer è possibile utilizzare **altri tipi di consumer** di dati
  - Tool che supportano Kafka versione 3.3.1
  - Software sviluppato ad hoc usando librerie che parlano con Kafka i.e. python

```
from kafka import KafkaConsumer
from json import loads

consumer = KafkaConsumer(
    '<topic-name>',
     bootstrap_servers=[<endpoint list>],
     auto_offset_reset='from-beginning',
     enable_auto_commit=True,
     security_protocol="SASL_SSL",
     group_id='test_group',
     sasl_plain_username='<username>',
     sasl_plain_password='<password>',
     sasl_mechanism='PLAIN',
     ssl_cafile='/pa-to-ca-
     certificate/ca.pem',
     value_deserializer=lambda
     x:loads(x.decode('utf-8')))

for message in consumer:
    message = message.value
    print(message)
```

Consumer group

- Consumer organizzati in *consumer group*
  - I dati di un topic vengono consumati **una sola volta per ogni *consumer group***
  - Esiste un offset per topic e per consumer group, in caso di necessità l'admin lo può spostare indietro

```
GROUP      TOPIC        PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG     CONSUMER-ID  HOST
CLIENT-ID
logstash   log-storage  0          1761235739      1776250685      15014946   <id>       /131.154.192.168
logstash-0
logstash   log-storage  2          1723651500      1738115448      14463948   <id>       /131.154.192.168
logstash-0
logstash   log-storage  1          1079491783      1094781926      15290143   <id>       /131.154.192.168
logstash-0
```

Opensearch - overview

- Strumento per aggregare, visualizzare e analizzare dati di vari tipi
  - Fork open source di Elasticsearch gestito da Amazon
  - Plugin analoghi a quelli forniti a pagamento con Elasticsearch
  - **Plugin Security** per la gestione di tenant, ruoli, utenti, permessi
- Configurata in HA (tre nodi master e quattro nodi data)
- Totale spazio di archiviazione 8 TB (SSD)
- Dopo una certa retention, i dati possono essere spostati su **repo Ceph** di dimensione per ora di 16 TB, ma facilmente estendibile

Opensearch - concetti chiave

- Dati organizzati in **indici**
  - Con criteri diversi, per ora divisi sempre per reparto e data (es. servnaz-2023.03.28)
  - Possibilità di creare **index pattern** per navigare o utilizzare dati contenuti in certi gruppi di indici
    - Ad es.: storage*, storage-2023.03*, storage-hsm*, farming*
  - Gli indici contengono **documenti** che corrispondono alle entry dei log e metadati (host da cui provengono, tag, eventuali campi costruiti da Logstash)
  - Vengono divisi in **shard** e replicati
  - Possibilità di applicare **index policies** (azioni eseguite in base al verificarsi di una condizione)
  - Possibilità di fare **snapshot** per tenere backup o mettere offline indici oltre una certa retention

Opensearch - autenticazione/autorizzazione

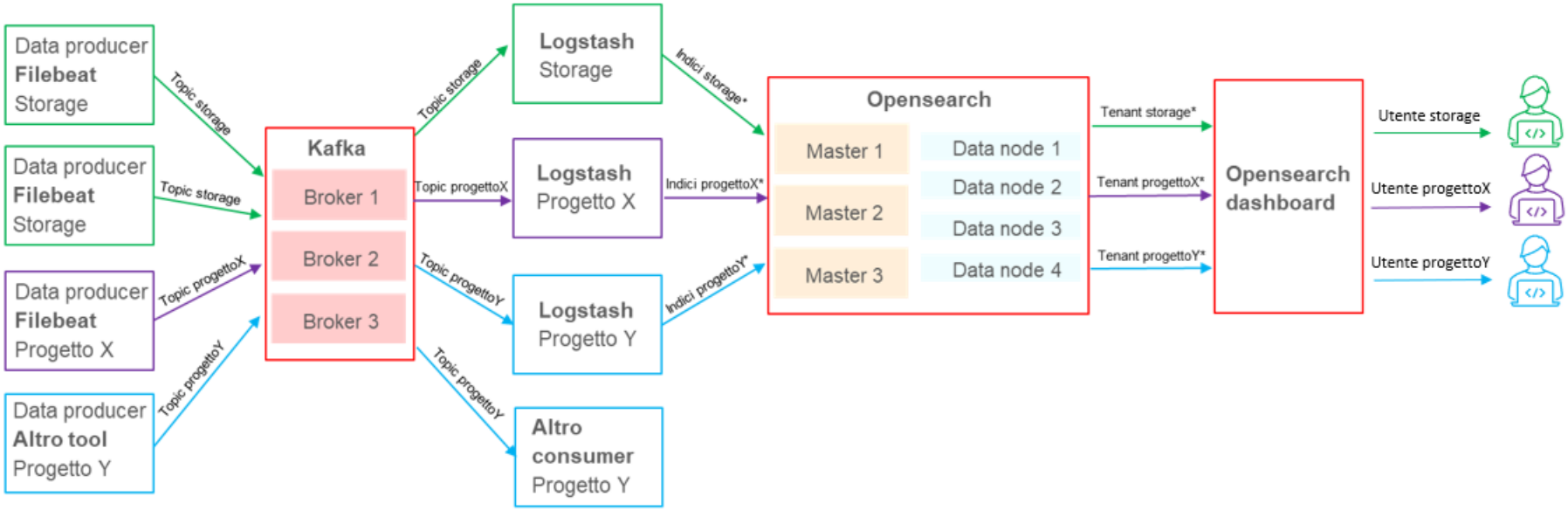
- Doppio metodo di autenticazione
  - **Locale** : user e password (admin e utenti di servizio)
  - Via **IAM CNAF** (iam.cnaf.infn.it)
    - Utenti locali IAM
    - Utenti **AAI INFN**
- Autorizzazione tramite **gruppi IAM**
  - I gruppi IAM dell'utente vengono mappati in automatico su *backend roles* di Opensearch
  - Ad ognuno di essi sono associati permessi per operazioni sul cluster Opensearch o sugli indici
  - Al momento gruppi dei reparti che usano la piattaforma
    - storage, farming, servnaz
    - gruppi con permessi read-only sui dati di ciascun reparto

Wazuh

- Piattaforma open source basata su OpenSearch
- Specifica per monitorare e allarmare in caso di eventi legati alla **sicurezza** e di incidenti
- Possibilità di effettuare azioni automatiche in caso di minacce
- Possibile integrazione nella piattaforma BDP
  - Versione 4.4 dovrebbe supportare OpenSearch, rilasciato oggi!
  - Dashboard separata che visualizza dati gestiti dal servizio OpenSearch

Wazuh - Docs

Schema piattaforma



Autenticazione / autorizzazione

- Comunicazione via **SSL** tramite certificati Puppet su tutti i nodi
  - Intra-cluster (Kafka, OpenSearch)
  - Tra servizi diversi
- Autenticazione / autorizzazione su Kafka
  - Producer e consumer (tra cui Logstash) si autenticano tramite utenti locali su Kafka
  - **ACL** sugli utenti per permessi sui topic
- Autenticazione su OpenSearch
  - Admin e utenti di servizio sono locali
  - Accesso utente tramite **IAM** (via browser e API)
- Autorizzazione su OpenSearch
  - Tramite permessi e ruoli (**security plugin**)

Scalabilità

- Piattaforma **costruita per poter scalare** orizzontalmente in maniera semplice e modulare
- In caso di risorse virtuali insufficienti (CPU / memoria)
  - Aumento risorse
  - Aggiunta nodi nei cluster Kafka / OpenSearch
- Auspicabile un server Logstash per reparto
  - Anche per gestire in autonomia i filtri dei propri dati
- In caso di spazio insufficiente su Kafka/OpenSearch
  - Acquisto nuovo hardware
  - Da valutare a giugno, dopo qualche mese di utilizzo

HA dei servizi e ridondanza dei dati

- Kafka e OpenSearch cluster **multinodo**
- Logstash per ora è unico
  - E’ tollerabile una indisponibilità fino a 1 settimana
- I dati su Kafka sono replicati tre volte, sui dischi di tre nodi diversi
- Gli indici su OpenSearch
  - Sono di default in **replica 2**
  - **Snapshot** periodiche degli indici di servizio (definizioni dashboard, configurazione ruoli, permessi, etc.) su storage Ceph
  - Possibile fare snapshot di indici di dati rilevanti su storage Ceph

Retention e archiviazione

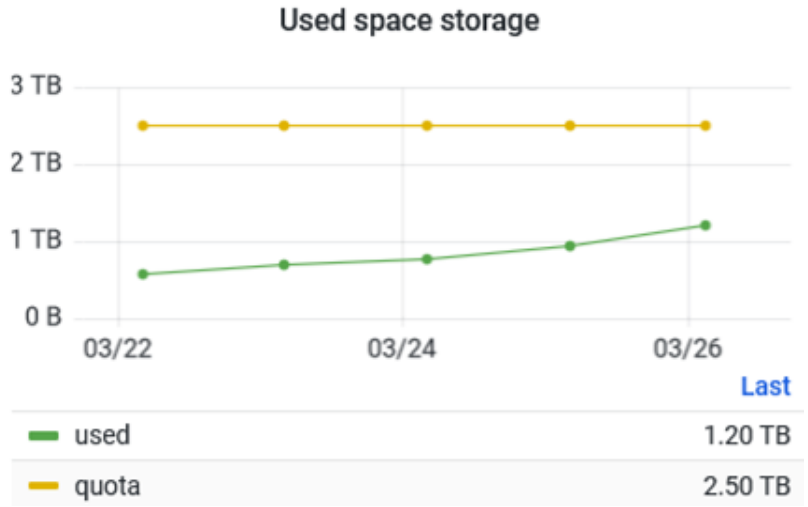
- Su Kafka una diversa **retention per topic**
  - Tipicamente una settimana
- Su OpenSearch
  - **Indici aperti**: occupano spazio sugli SSD e altre risorse
  - **Indici chiusi**: occupano spazio sugli SSD, ma non altre risorse
  - **Indici offline**: occupano spazio su storage Ceph remoto
  - Possibilità di chiudere / mettere offline gli indici sulla base dell'età, dello spazio occupato, del numero di record contenuti (tramite index policies)
- Infrastruttura dimensionata per tenere una settimana di indici aperti, un mese di indici chiusi, sei mesi di indici offline
  - Reali retention verranno stabilite considerando le reali dimensioni dei dati
  - Effettuati test delle procedure per corretta gestione

Multi-tenancy

- In OpenSearch ciascun utente può accedere a:
  - Un tenant **pubblico** comune a tutti gli utenti
  - Un tenant **privato** non accessibile da altri
  - Uno o più tenant di **reparto/progetto** con dashboard specifiche (se possiede i gruppi IAM che ne permettono l'accesso)
- Ogni utente può agire sui dati del proprio reparto/progetto
- Possibilità di accesso read-only ad altri dati

Segregazione dei dati

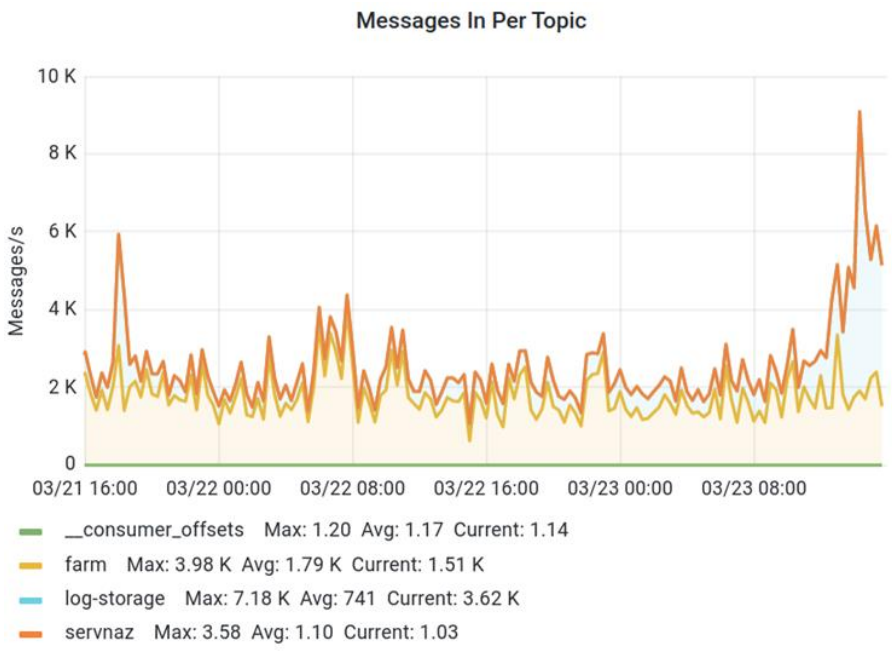
- **Segregazione** dati per reparto / progetto
  - Ogni reparto può scrivere/leggere solo sul/dal proprio topic Kafka e accedere ai propri indici OpenSearch
  - Possibili permessi di lettura su altri dati
- Gestione **quote** non presente in OpenSearch
  - Sviluppata utility che controlla spazio occupato da indici di ciascun reparto e lo confronta con una quota
  - Notifica via mail in caso di soglia raggiunta

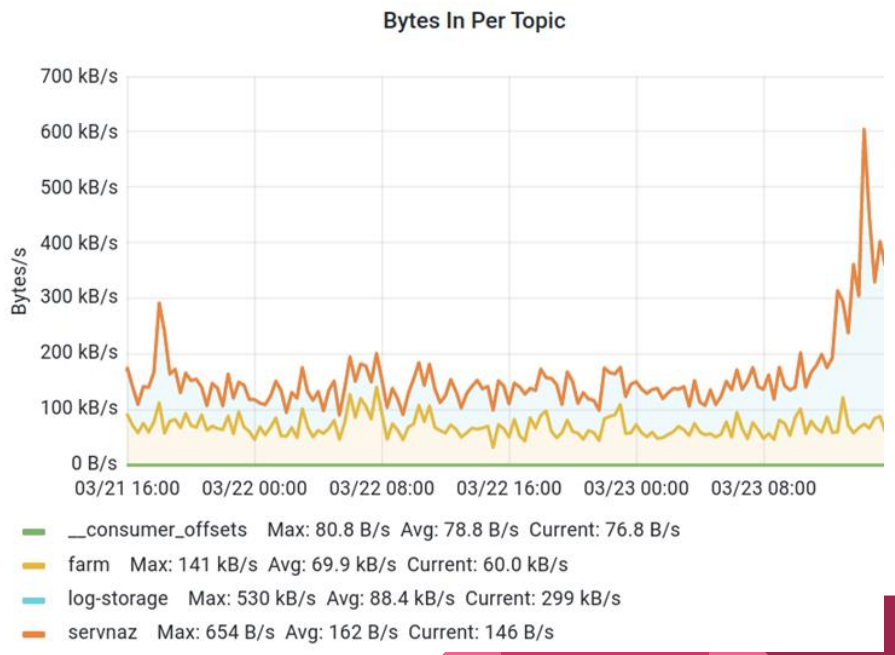


Integrazione con servizi esistenti

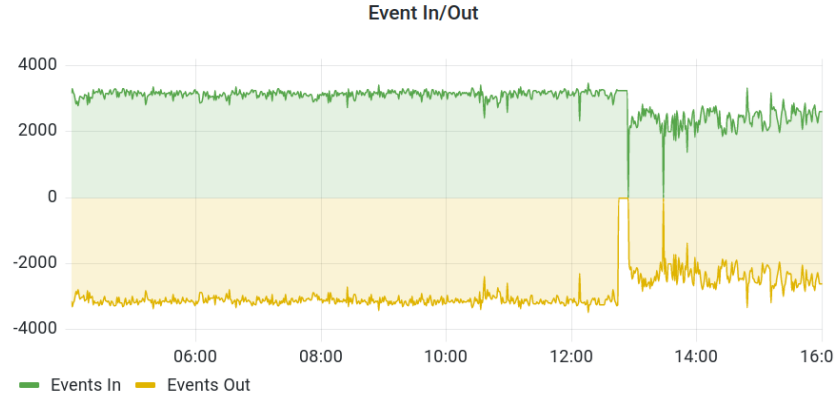
- Tutti nodi integrati in **Foreman/Puppet**
  - Moduli Puppet dei vari servizi da fare
- Autenticazione /autenticazione tramite **IAM** cnaf
- **Monitoring** con servizi CNAF
  - Sensu / Influx / Grafana
  - Check su stato dei servizi, occupazione spazio
- **Documentazione** su Confluence
  - Descrizione infrastruttura
  - Doc per utenti
  - Doc per amministratori
  - Per ora su sezione CNAF Reloaded

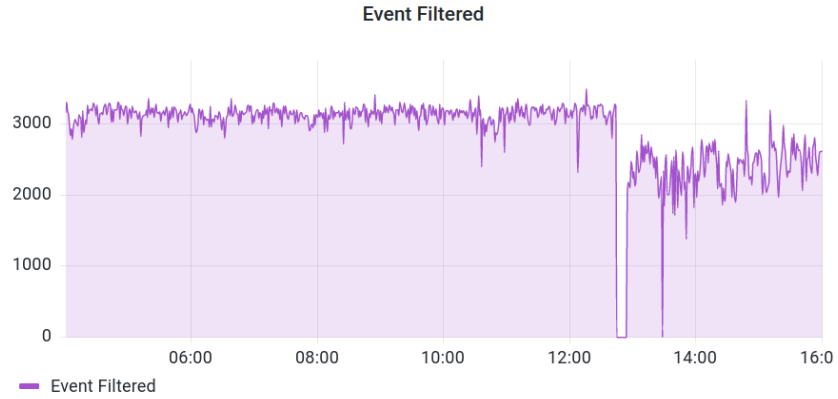
Monitoring - Kafka



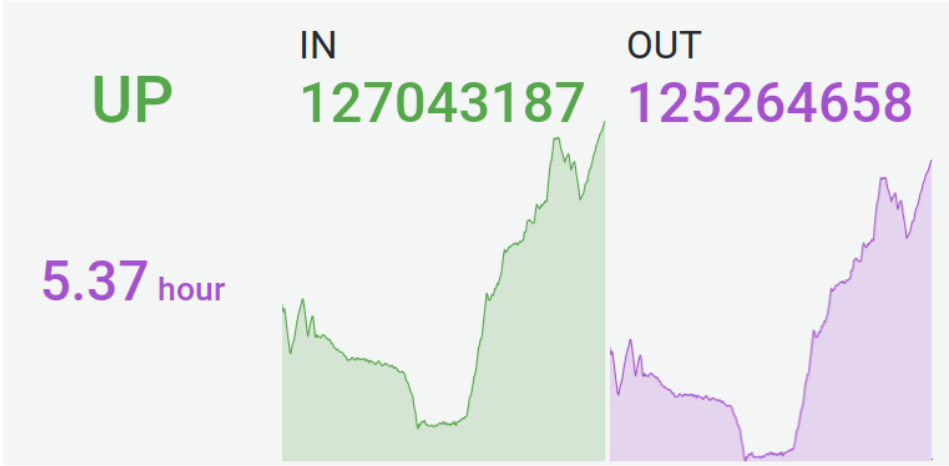


Monitoring - Logstash





Last 6h



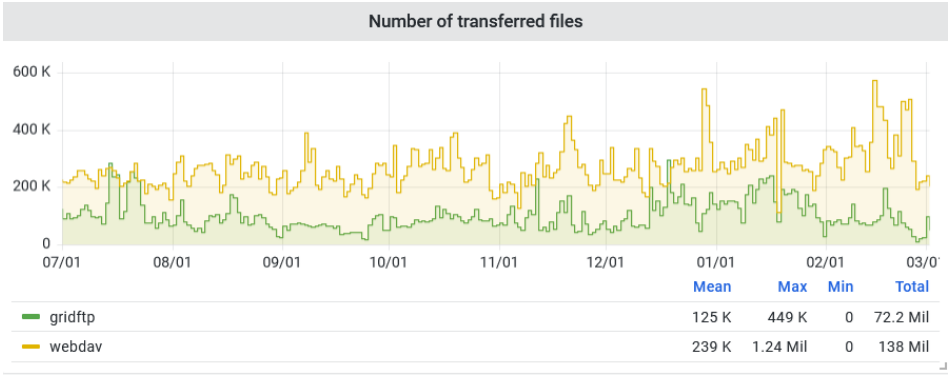
Monitoring - Opensearch

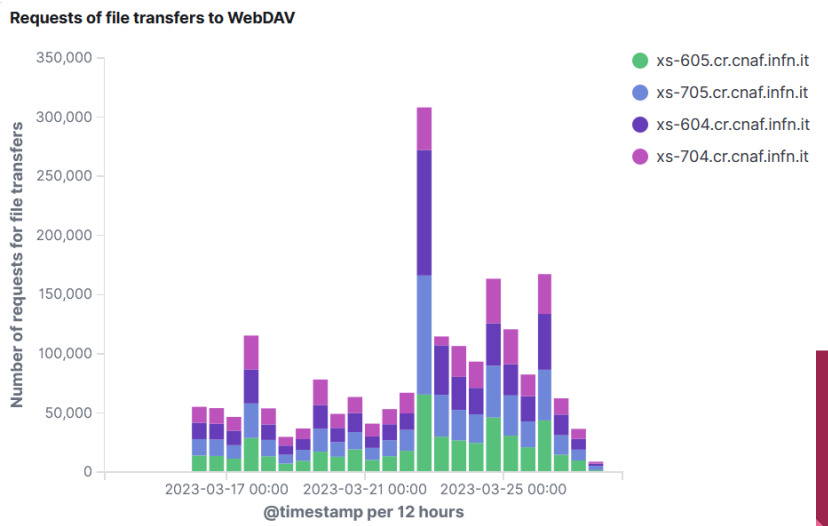
Demo 1

- Kafka
  - Creazione topic
  - Settaggio ACL
- Producer python
  - Verifica che arrivino dati a Kafka
- Consumer python
  - Verifica che i dati vengono consumati da 2 consumer (con e senza consumer group)
- Dashboard OpenSearch
  - Accesso, ruoli, tenant
  - Index pattern
  - Discover, con possibilità di filtrare (ad es. file di VO da cercare in log di vari servizi)
  - Query SQL-like
  - Uso di API da dashboard
- Uso API via command line (con token IAM)

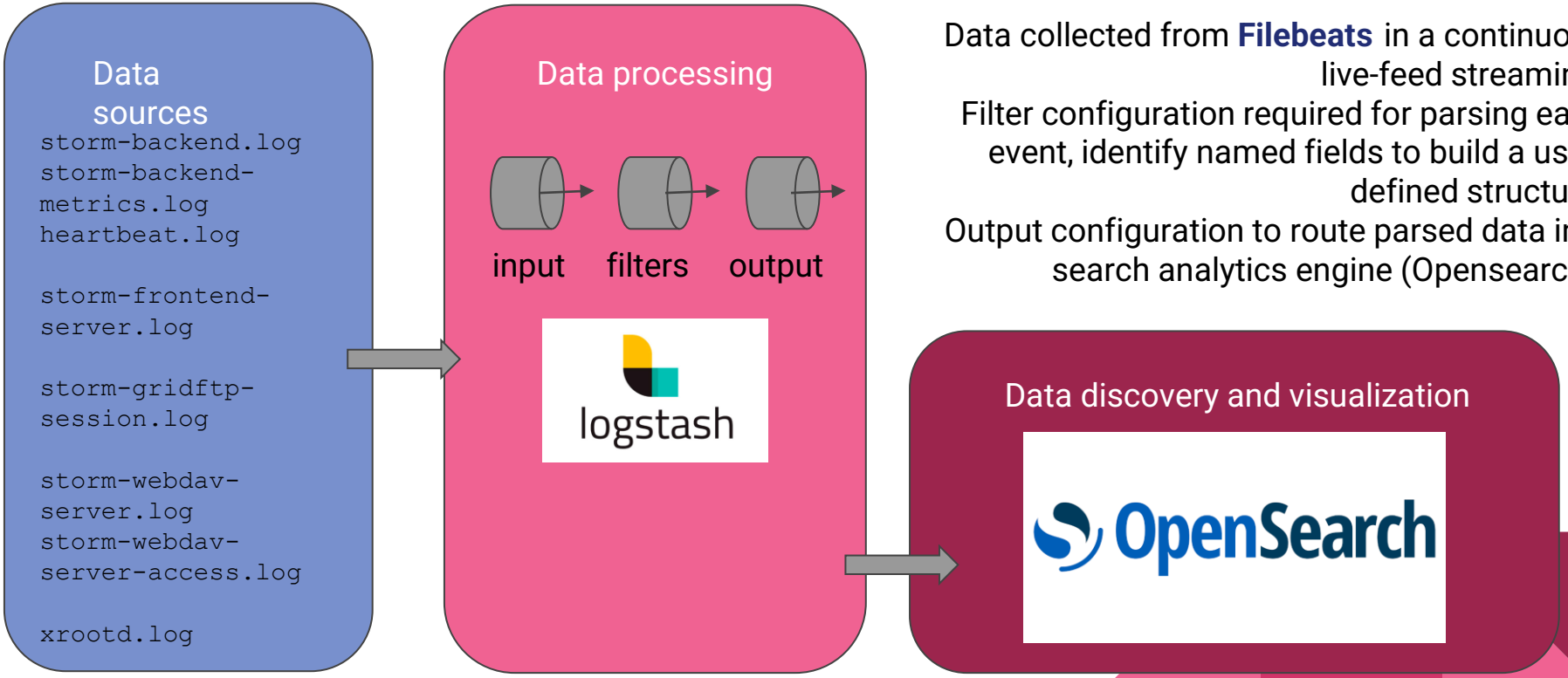
Demo 2: dai log ai plot

- Punto di partenza: il lavoro di Tommaso Diotallevi e Luca Giommi (2018)
- La “BDP” di storage (stack ELK), in produzione dal 2019, molto utile nel monitoring dei servizi (a integrazione di t1metria) e delle varie data challenge, e recentemente copiata sulla nuova infrastruttura



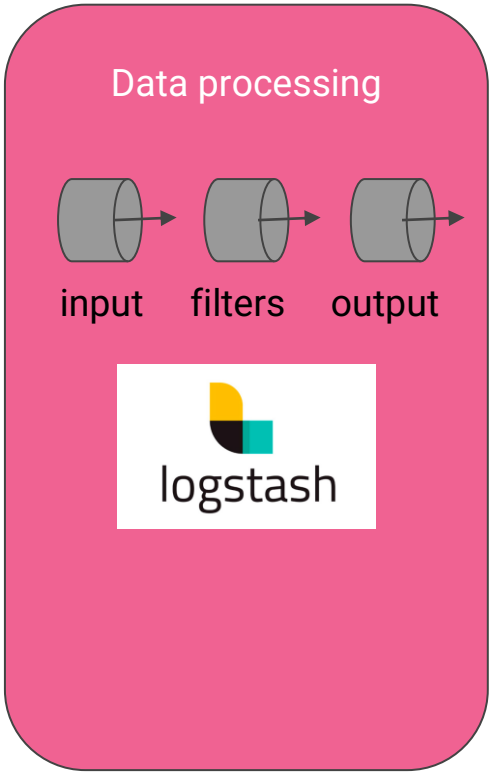


Demo 2: log parsing con Logstash



Data collected from **Filebeats** in a continuous live-feed streaming.
Filter configuration required for parsing each event, identify named fields to build a user-defined structure.
Output configuration to route parsed data in a search analytics engine (Opensearch).

Demo 2: log parsing con Logstash



Grok Debugger | Debugger | Discover | Patterns

2020-07-16T19:20:30.45+01:00 DEBUG This is a sample log

%{TIMESTAMP_ISO8601:time} %{LOGLEVEL:logLevel} %{GREEDYDATA:logMessage}

☐ Add custom patterns ☐ Keep Empty Captures ☑ Named Captures Only ☐ Singles ☐ Autocomplete Go


```
{
  "time": [
    [
      "2020-07-16T19:20:30.45+01:00"
    ]
  ],
  "logLevel": [
    [
      "DEBUG"
    ]
  ],
  "logMessage": [
    [
      "This is a sample log"
    ]
  ]
}
```


Log data are parsed using **grok** filter based on Regular Expressions, matching specific portions of log entries by creating a series of pattern defined as:
%{PATTERN:Field_Name}
where PATTERN is the name of the pattern that will match the text, while Field_Name is the identifier of the piece of text being matched.

Demo 2

- Dashboard OpenSearch
  - Esempio di log parsed con Logstash su OpenSearch (**Discover**)
  - Esempio di visualizzazione delle informazioni dei log (**Visualize**)
  - Creazione dashboard e dashboard esistenti per servizi data management e data transfer (**Dashboard**)

Stato e prossimi passi

- Infrastruttura utilizzabile per gestione dati di reparti/progetti CNAF
  - In varie modalità, come mostrato
- Già presenti dati di diversi host di storage, farming e servnaz
- Integrazione con Wazuh non appena disponibile la versione 4.4
- Da fare
  - Moduli Puppet dei vari componenti (Kafka, Logstash, OpenSearch)

Link e contatti

- **Documentazione generale sulla piattaforma:** https://confluence.infn.it/display/CR/Descrizione+dell%27infrastruttura
- **Documentazione per utenti**: https://confluence.infn.it/pages/viewpage.action?spaceKey=CR&title=Documentazione+per+utenti
- **IAM CNAF**: https://iam.cnaf.infn.it
- **OpenSearch dashboard**: https://cnlog-osdashboard01.cr.cnaf.infn.it
- Per info e richieste: bdp-deploy@lists.cnaf.infn.it