PunchPlatform Storage Architecture

PunchPlatform team

Agenda

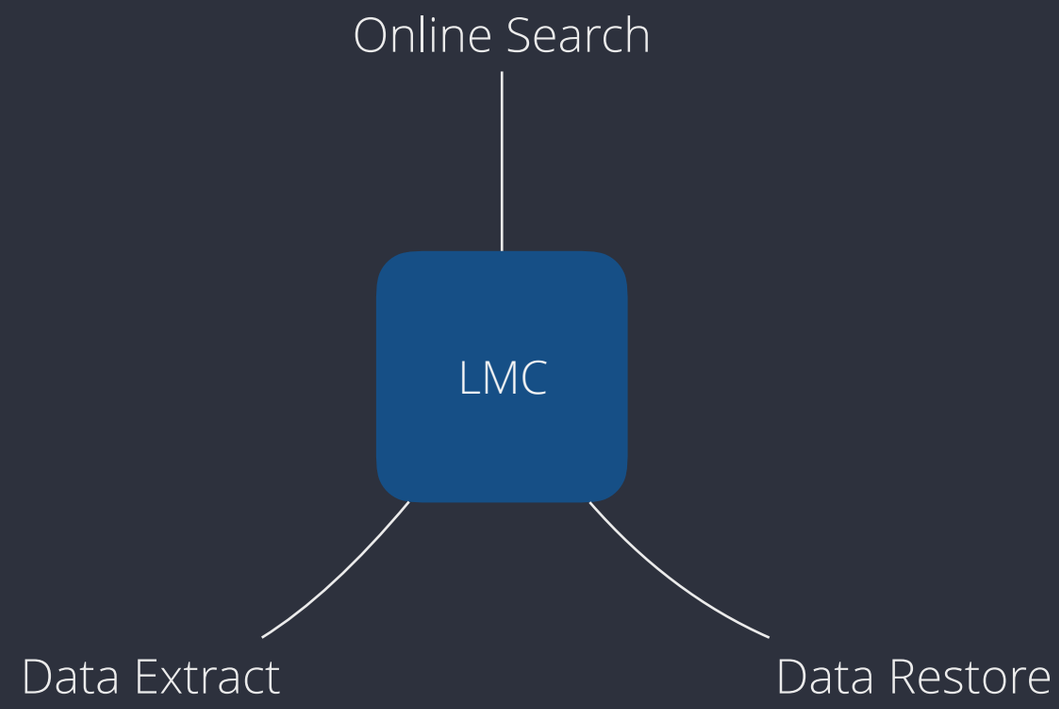
- How It Works
- Rationale
- Thanks

Logs matter. A PunchPlatform LMC is in charge of the following key services:

- storage of year(s) of logs
- online search over several months of logs
- efficient search over year(s) of logs
- keep logs usable to arbitrary processing
- on demand extractions

In this presentation we focus on the storage architecture : is it safe ? Can you loose a log ? It is expensive ?

In short : how does the PunchPlatform fulfil these requirements ?



A LMC is deployed on a farm of servers. Each equipped with local storage. No San/Nas required, no NFS.

The PunchPlatform relies on a modern scalable and resilient storage architecture, the one used in cloud and big data computing.

As an example : each server is equipped with :

- 46 To data on each server
- 24 disks 2To each, RAID-5 (n - 1)

That only provides local storage with RAID-5 protection to deal with the loss of a disk unit.

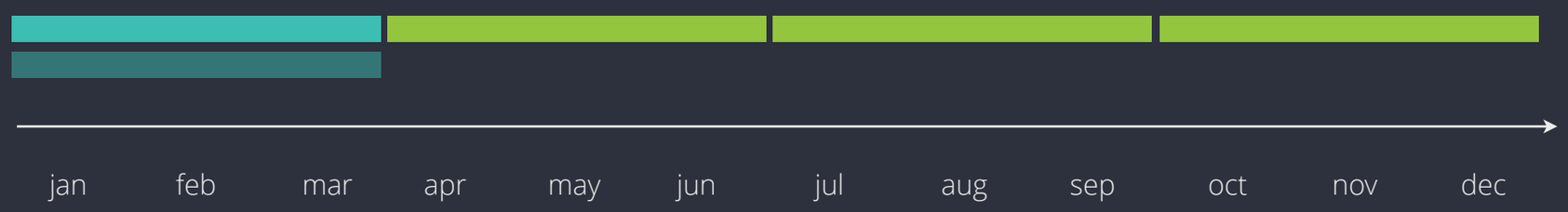
The PunchPlatform provides additional data replication as explained next.

Servers

Storage for Search Services

To provide online search, an *ElasticSearch* cluster is deployed over the servers. ElasticSearch is granted a fraction of the total storage. In addition, the search service is configured in two time periods corresponding to two different service level agreements:

- ***critical online period*** : searching over the first few months of data is a critical service. That service must survive server failure *with no service interruption*.
- **long-term online period** : the last (i.e. 9) months of logs must be quickly searchable, to perform specific forensics on past logs. It is acceptable to make that service temporarily unavailable after a server failure, as long as the data can be restored.



1-3 months are replicated : Each log is stored on 2 (or more) servers.
4-12 months are not replicated : you loose a server you loose your log.

where you fix the cursor >< is your decision

Here is a physical view: internally, ***ElasticSearch*** is in charge of :

- dispatching the logs to all servers (through indexes *sharding*)
- replicating each period by making sure the required shards are replicated on two different servers.
- Make all these logs safely searchable.

PunchPlatform takes care of :

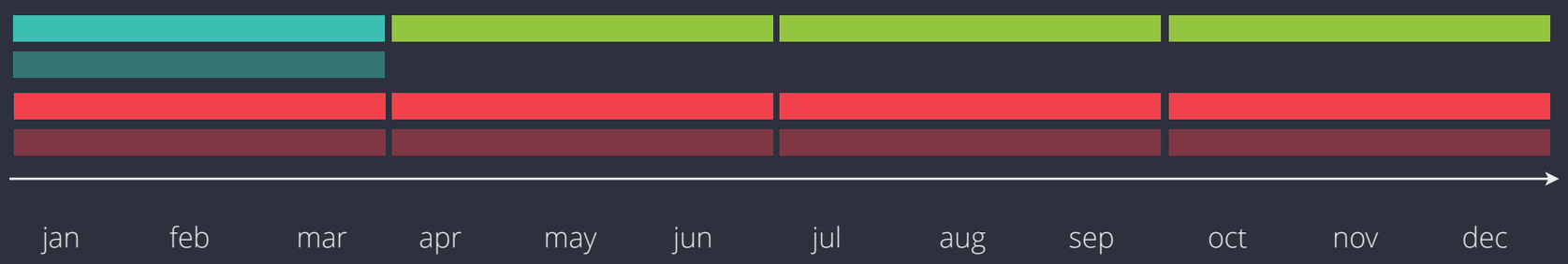
- automatic deployment and update
- multi tenancy
- housekeeping
- monitoring
- exposing the data to user(s)

Storage for Extract/Restore Services

Another need is to store a secondary copy of the logs, in order to efficiently provide on demand extracts, and/or to repopulate the long term search service after a server failure.

The PunchPlatform LMC provides an *Object Storage* service.

Before having a look at it, here is the principle : logs are kept for a year, using a sliding window scheme. Of course, logs are replicated so as to never loose some in case of severe disk/server failure.



1-12 months are replicated. Each log is stored on 2 (or more) servers.

Storage for Extract/Restore Services : **CEPH**

Storage for the Search Services

Here is the physical view. Internally a *CEPH* distributed object storage service is in charge of:

- dispatching the logs to all servers (through *Ceph object pools*). See it as files each containing a bunch of logs
- replicating logs. Several option here, one is the *Erasure Coding pools.* Instead of fully replicating the data to two servers, a RAID-5 like distributed algorithm is used.

Check out :

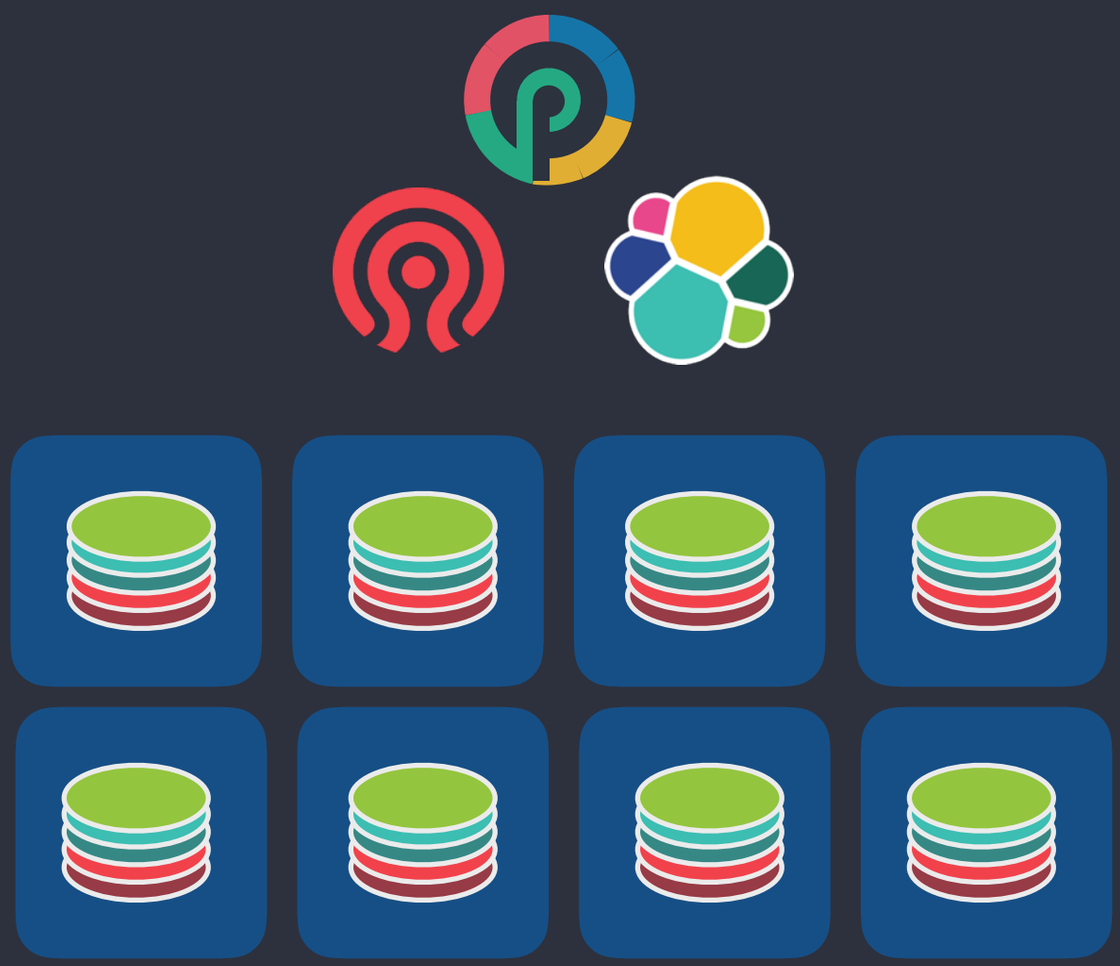
- https://ceph.com
- https://youtu.be/QBkH1g4DuKE
- https://www.redhat.com/fr/files/resources/en-rhst-ceph-rapid-vs-replication-INC0220894.pdf

Putting It All Together : the PunchPlatform

These two PunchPlatform services are deployed on the same physical servers/disks.

This architecture provides the best cost-to-performance ratio : at the end you need a fair amount of disk (of course), but also of RAM and CPU : the real service to you is to perform complex queries over months of data, taking full advantage of ElasticSearch indexing power. Some of your queries *will* need a lot of resources.

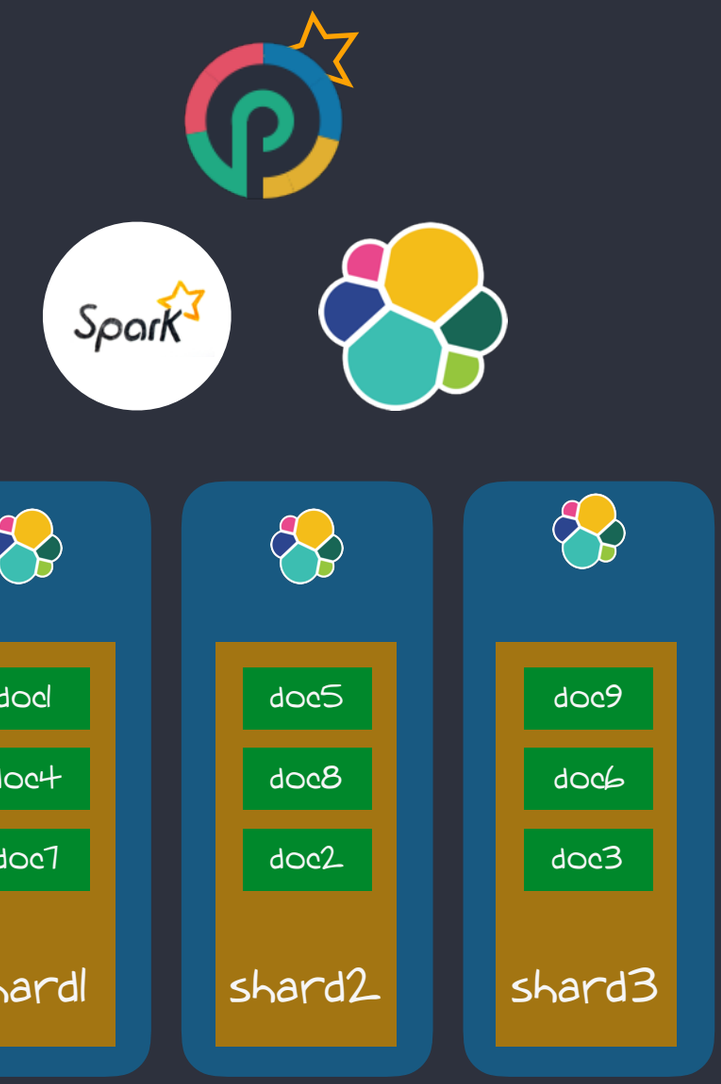
Using ElasticSearch alone is not a good fit for massive extractions, using CEPH alone is not a good fit for indexed search. What you need is a smart combination of the two technologies. That is in PunchPlatform.



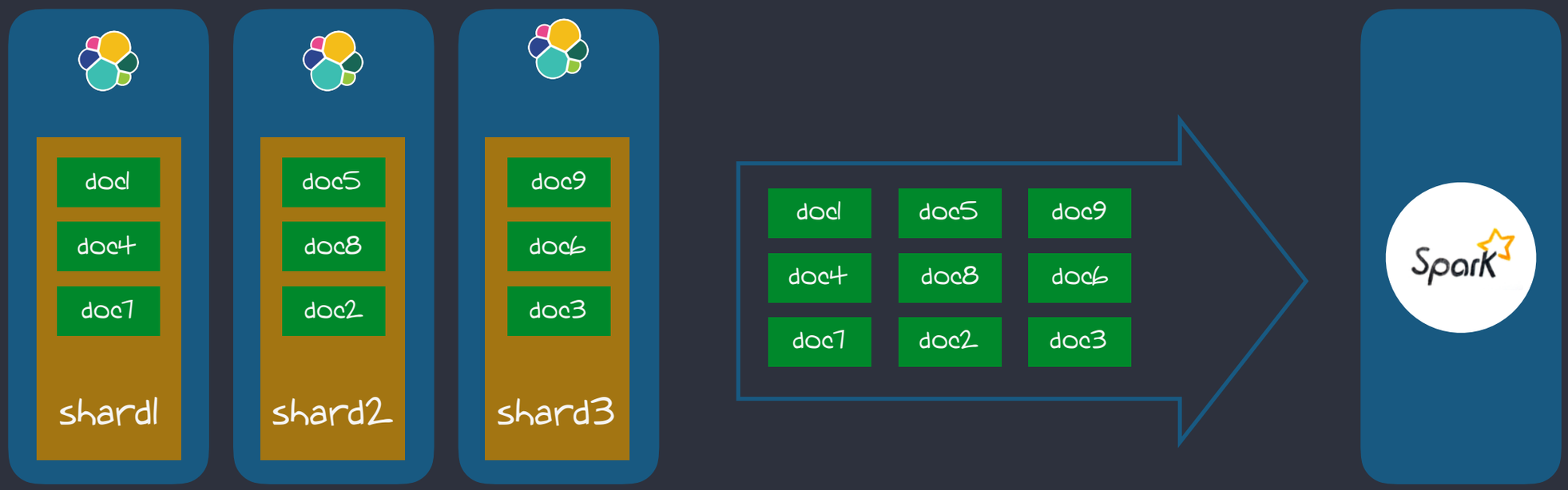
These two PunchPlatform services are deployed on the same physical servers/disks.

This architecture provides the best cost-to-performance ratio : at the end you need a fair amount of disk (of course), but also of RAM and CPU : the real service to you is to perform complex queries over months of data, taking full advantage of ElasticSearch indexing power. Some of your queries *will* need a lot of resources.

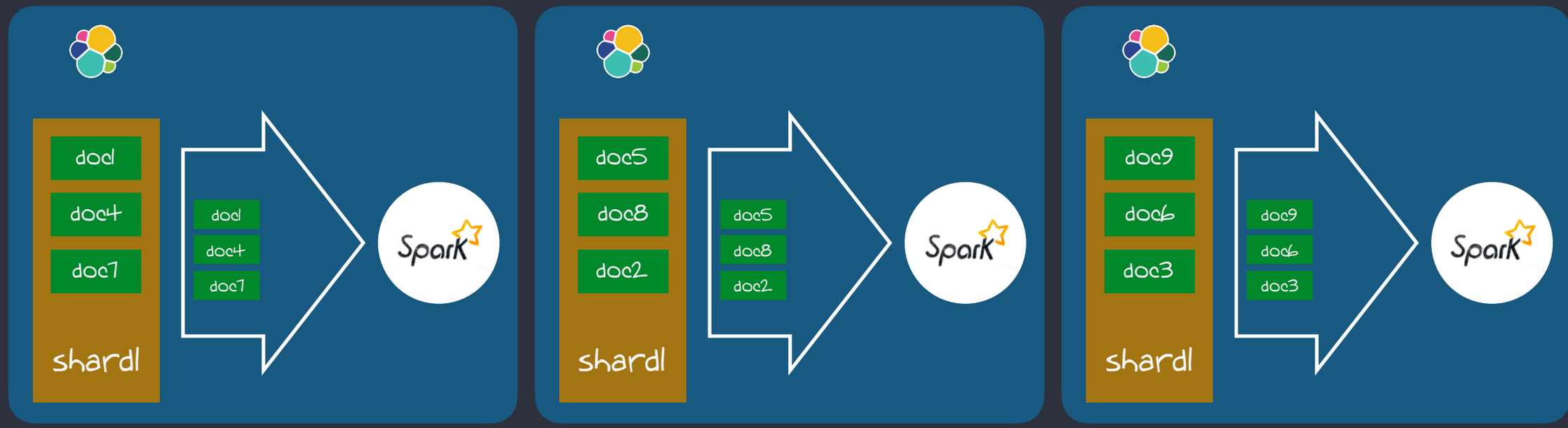
Using ElasticSearch alone is not a good fit for massive extractions, using CEPH alone is not a good fit for indexed search. What you need is a smart combination of the two technologies. That is in PunchPlatform.



Putting It All Together : the PunchPlatform



Putting It All Together : the PunchPlatform



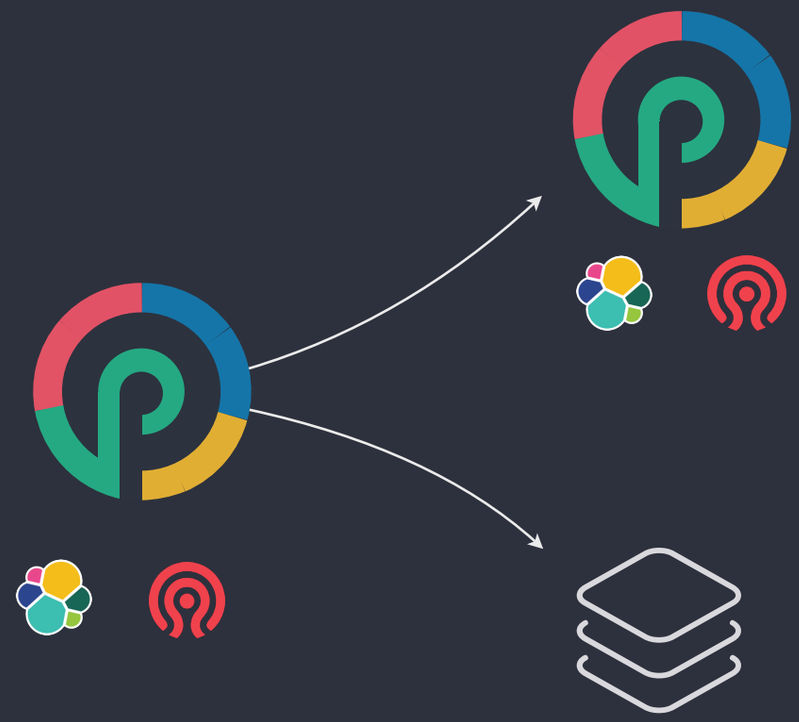
Is It Enough ?

Say you want to be protected against :
- site failure
- human attacks

Keep one more copy on an external remote system:

- a remote PunchPlatform: using the resilient *PunchPlatform DataFlow* transport module.
- Third-Party File Systems : using the PunchPlatform file export module.

This architecture provides asynchronous data replication. You select the data to be saved:
- raw logs
- enriched logs



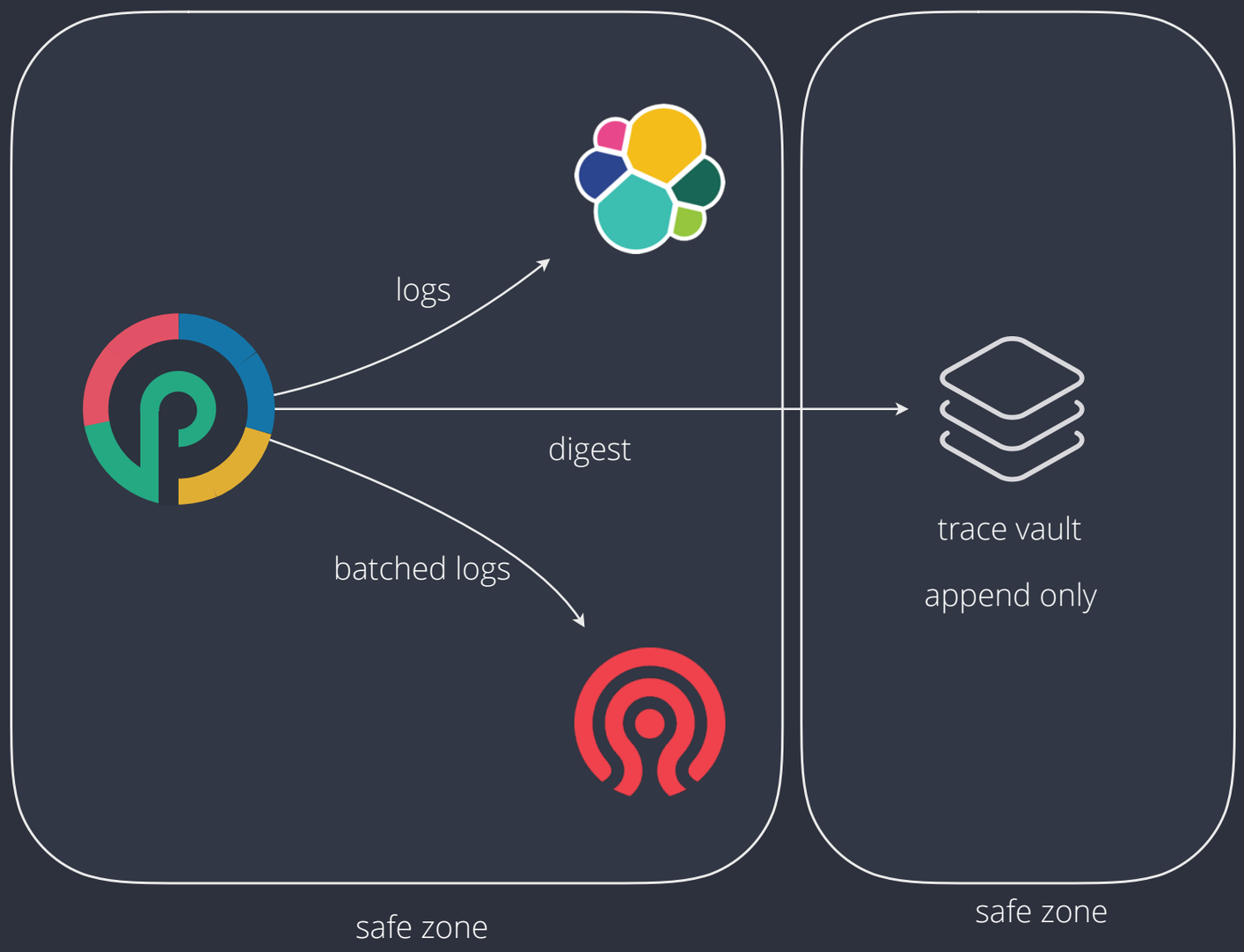
Integrity Control

Batches of logs are computed together with a corresponding hash (#).

- Logs are appended to the CEPH backend.
- a digest message including a hash (SHA-256) is forwarded to a remote append only *trace vault*, deployed and operated in a distinct security zone.
  - the hash is based on log content and timestamp
  - The third party trace vault provides the necessary level of protection.

Corrupting the data requires :

- accesses to distinct security zones
- defeat the trace vault policy



Does It guarantee data integrity ?

In the Data Stream :

All components in a LTR/LMR/LMC pipeline process logs in real time :

- hard to compromise a component before the corresponding traces are indexed/saved and protected downstream
- this requires all system access be protected with adequate logging forwarded to LMC

On the backend side :

- Every update or removal of logs from an ElasticSearch index or from an archive log file is detectable.
  - Assuming the attacker has no access to both security zones
- Signing method substitutions and log files a posteriori insertions are prevented by the digest information content.

In case of audit :

- The integrity checking of the archived logs can be achieved by recomputing the hashes from the archive space, and comparing them with the hashes from the trace vault.
- Accessing the saved logs can be achieved by replaying the logs from the CEPH archive back into an ElasticSearch cluster.

This makes it possible to confirm the integrity of possibly large periods of logs.

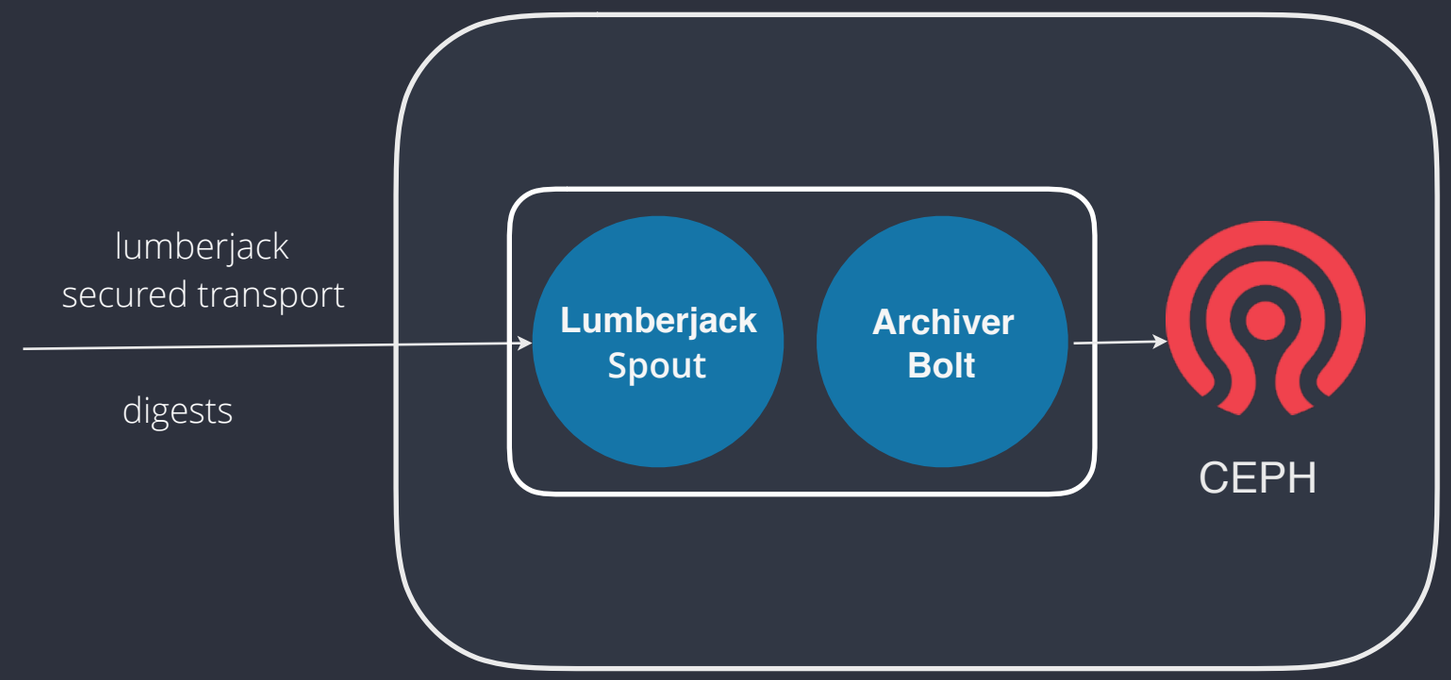
Implementing a TraceVault on top of PunchPlatform

An easy solution is to design a trace vault on top of the PunchPlatform.

- Lumberjack as acknowledged transport protocol.
- One or three nodes depending on the reliability requirements
- Many possibilities for saving the signatures :
  - **CEPH**
  - Files
  - ElasticSearch
  - Kafka

Recommendation :

- this is a critical service
- use a 3-node punch platform cluster.

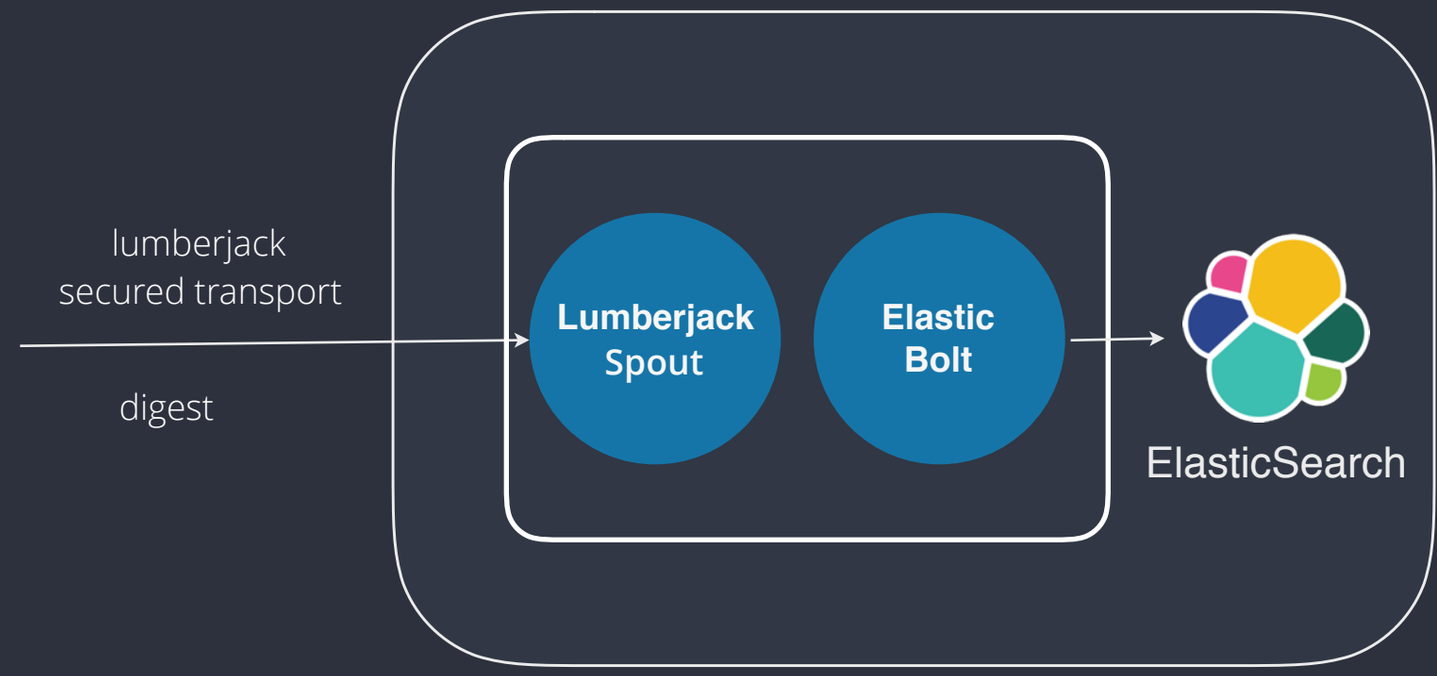


Implementing a TraceVault on top of PunchPlatform

A solution to provide an easier access and selection of digests : design a trace vault on top of ElasticSearch.

- Kibana based searching and extraction.

The Punchplatform is highly configurable : several backends can be used altogether.



Thanks !