Elasticsearch, Loki, Sentry: 3 outils pour une approche DevOps

Jonathan Schaeffer, Simon Panay

2025-05-26

Contexte

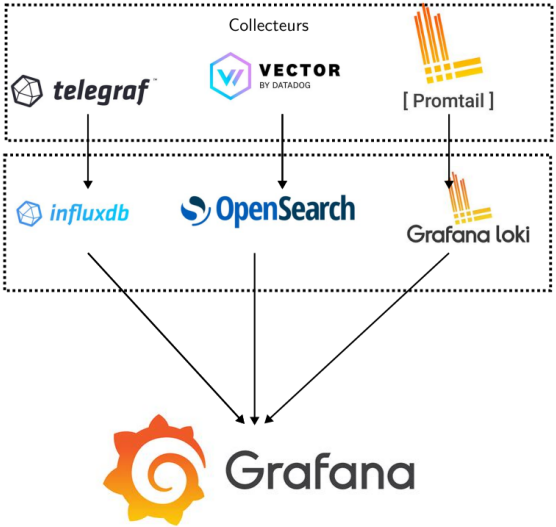
Le centre de données sismologiques Epos-France

- le centre de données livre les données et métadonnées par webservice (requêtes HTTP), $\sim$ 8 req/s
- pour un même service, plusieurs briques logicielles sont activées
- les actifs logiciels sont hétérogènes (sur étagère, à façon, de différentes époques)
- les logs sont éparpillés, et hétérogènes
- nous mettons en place du DevOps

Les problématiques opérationnelles

1. extraire des indicateurs de performance des services
2. après un incident
   - retrouver facilement tous les logs pertinents
   - analyser les logs pour extraire de l'information
   - croiser les logs avec des métriques systèmes

Architecture



Notre fonctionnement avec opensearch

Traitement d'un log

Le pipeline saucissonne un log et en extrait des paires de clés/valeurs spécifiques.

```
May 20 11:36:43 resif-rproxy haproxy[3661045]: 194.73.165.245:59414 [20/May/2025:
prod~ resif-k8s/w08
0/0/1/16/17 204 191 - - ---- 6/6/1/0/0 0/0
{ws.resif.fr|ObsPy/1.2.2 (Linux-5.14.21-150500.55.73-default-x86_64-with-glib} {}
"GET /fdsnws/dataselect/1/query?starttime=2023-07-07T05%3A30%3A06.000111&endtime=
07-07T06%3A20%3A06.000111&network=G&station=STPA&location=00&channel=BHZ HTTP/1.1
```

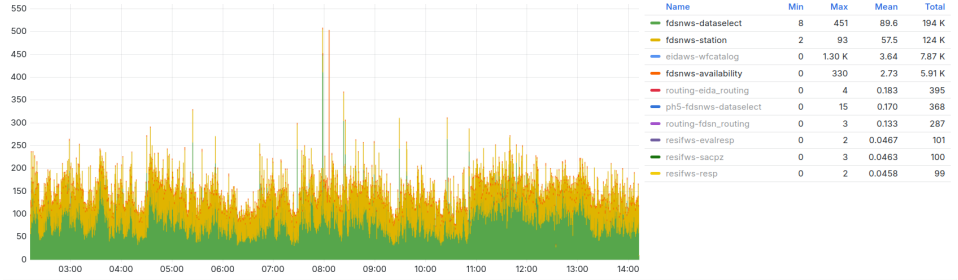
Attributs supplémentaires récupérés

service.fullname fdsnws-dataselect
useragent.name ObsPy

La recherche se fera sur des valeurs de certains champs, de préférence indexés (des "termes").

Grafana et opensearch

Exemple de graphs:



Exemple de post-incident

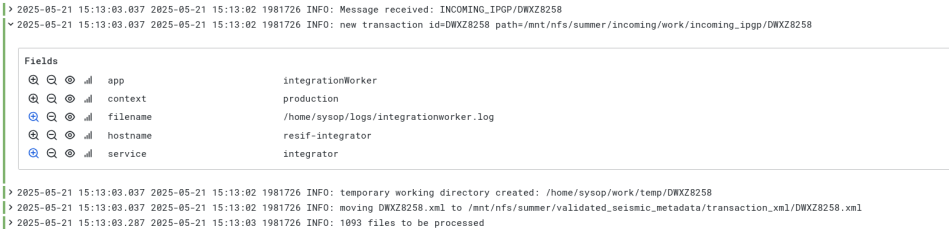
- Mardi dernier (20/05), un webservice s'effondre, se remet en place, s'effondre à nouveau. Après intervention, on veut comprendre ce qui s'est passé.
- A-t-on une anomalie dans les flux de requêtes ?
- Quels indicateurs pourraient nous renseigner ?
  - nombre de requête par service
  - nombre de requête par user agent
  - nombre de requête par origine
  - contenu de la requête
  - consommation mémoire de la base de données

Lien vers le dashboard
Exploration des logs par attribut Vidéo
Croisement avec des métriques prometheus: Vidéo

Notre fonctionnement avec Loki

Principe de fonctionnement

Loki n'analyse pas les logs, il les stocke et y attache des labels (clés+valeurs).
L'agent promtail observe les fichiers de logs (ou autres flux), attache des labels configurés et envoie à Loki.
Par exemple:



La requête dans les logs (LOGQL) se fait sur les labels et les contenus de logs.

Extraire des infos des logs

Un exemple de scénario de recherche (Vidéo)
Pour du DevOps, c'est très intéressant car on a un outil partagé entre développeurs et opérateurs pour extraire de l'information.

Notre fonctionnement avec Sentry

Capture et alerte sur les exceptions logicielles

- Groupement d'erreur par exception/stacktrace (ou +, configurable)
- Pour chaque erreur :
  - liste des évènements problématiques
  - stacktrace et état des variables
  - entête http des requêtes
  - logs/requêtes SQL du contexte
  - etc...
- Intégration avec les issues Gitlab/Github

Démo en ligne

=> Gestion fine des incidents sans être "noyé" sous les logs

Application Performance Monitoring (APM)

- Métriques
  - requêtes SQL
  - requêtes vers API externes
  - requêtes vers serveurs de cache valkey/redis
  - queues de messages (airflow, celery, spark)
- Tracing
- Profiling

=> identification aisée des points de faiblesse d'une application

Conclusion

OpenTelemetry: de nouvelles perspectives

- Une spécification et un protocole ouverts pour l'observabilité des services.
- Une implémentation (SDK) pour beaucoup de langages
- Un service pour exposer les métriques, scrapables par Prometheus

Conclusion

- Les logs, c'est bien quand on peut les exploiter
- Un environnement DevOps, c'est mieux avec des outils communs.
- Cercle vertueux pour améliorer la qualité et la valeur des logs
  - l'opérateur exploite les infos des logs
  - le développeur comprend les infos utiles pour l'opérateur

texte pur.