



# Gestion des services Internet

Cours 4 - Le monitoring et le logging de services Web

Un cours de Yann Fornier



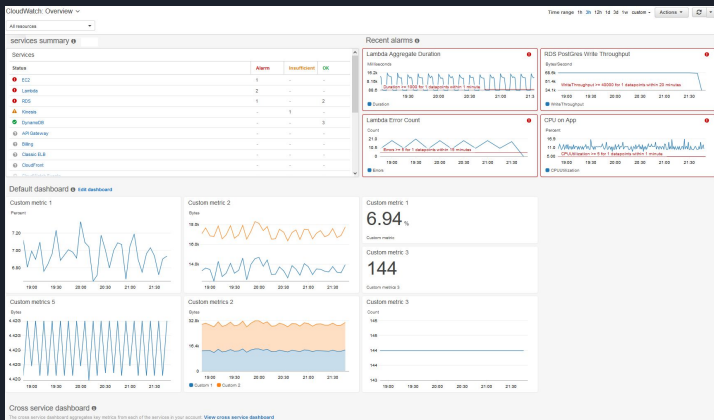
## Thèmes abordés

Mise en place de monitoring et de logging pour suivre les performances et la disponibilité des services web

Les différents outils et techniques utilisées pour le monitoring et le logging des services web

# Introduction

Le monitoring et le logging sont des pratiques clés pour assurer la disponibilité, la performance et la sécurité des services web.



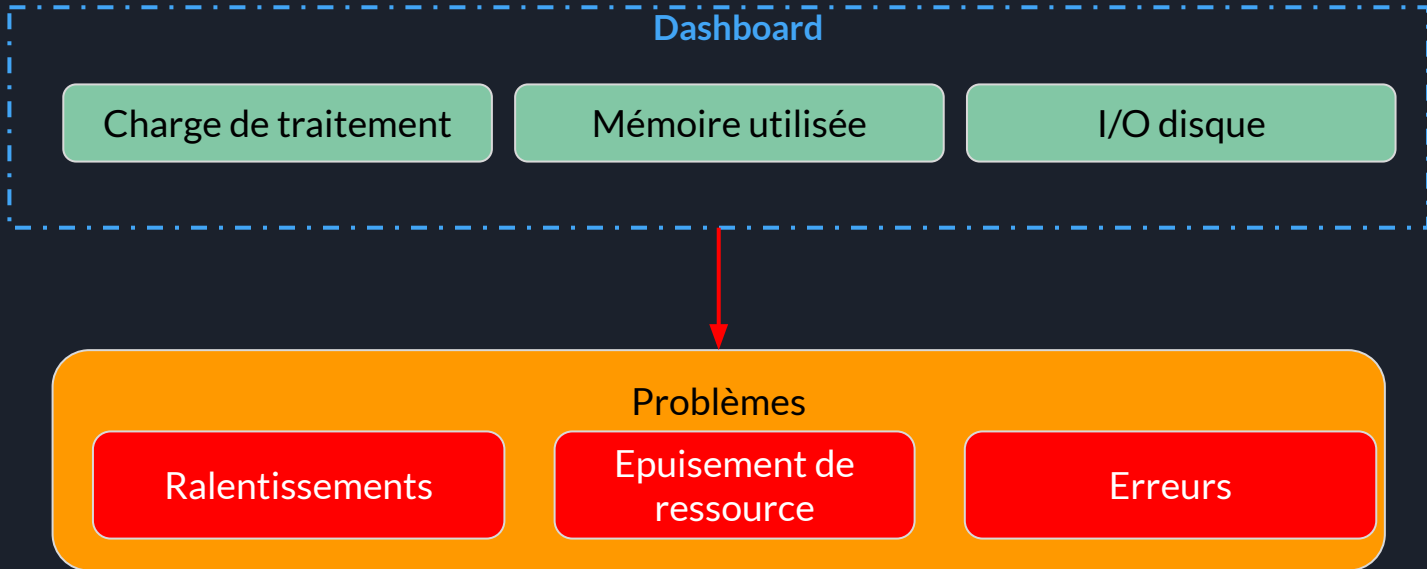
Monitoring

```
Nov 29 10:10:02 lograzer filebeat[21206]: 2020-11-29T10:10:02.278Z#011INFO#011[monitoring]#011log/lo
g.go:145#011Non-zero metrics in the last 30s#011{"monitoring": {"metrics": [{"beat": {"cpu": {"system":
{"ticks":167030}, "total":{"ticks":360860, "time":{"ms":3}, "value":360860}, "user":{"ticks":193830, "tix
e":{"ms":3}}}], "handles":{"limit":{"hard":4096, "soft":1024}, "open":12}, "info":{"ephemeral_id":"ec16b3
47-453b-416b-a781-5992dd598d34", "uptime":{"ms":118290072}}, "memstats":{"gc_next":52521504, "memory_a
lloc":27085744, "memory_total":3571799880}, "runtime":{"goroutines":493}, "filebeat":{"harvester":{"po
en_files":3, "running":2}}, "libbeat":{"config":{"module":{"running":0}}, "pipeline":{"clients":4, "even
ts":{"active":4118}}}, "registrar":{"states":{"current":4}}, "system":{"load":{"1":0, "15":0, "5":0, "non
m":{"1":0, "15":0, "5":0}}}}]}
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.122Z#011INFO#011[pipeline/output.go:10
]#011Failed to connect to backoff(elasticsearch(http://192.168.1.114:9200)): Get http://192.168.1.11
4:9200: EOF
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.122Z#011INFO#011[pipeline/output.go:93#
011]attempting to reconnect to backoff(elasticsearch(http://192.168.1.114:9200)) with 1570 reconnect
attempt(s)
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.123Z#011INFO#011[publisher]#011pipelin
e/retry.go:196#011retryer: send unsuit-signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.123Z#011INFO#011[publisher]#011pipelin
e/retry.go:198#011 done
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.123Z#011INFO#011[publisher]#011pipelin
e/retry.go:173#011retryer: send wait signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.123Z#011INFO#011[publisher]#011pipelin
e/retry.go:175#011 done
Nov 29 10:10:32 lograzer filebeat[21206]: 2020-11-29T10:10:32.278Z#011INFO#011[monitoring]#011log/lo
g.go:145#011Non-zero metrics in the last 30s#011{"monitoring": {"metrics": [{"beat": {"cpu": {"system":
{"ticks":167030, "time":{"ms":3}}, "total":{"ticks":360870, "time":{"ms":6}, "value":360870}, "user":{"ti
cks":193840, "time":{"ms":3}}}], "handles":{"limit":{"hard":4096, "soft":1024}, "open":12}, "info":{"ephe
meral_id":"ec16b347-453b-416b-a781-5992dd598d34", "uptime":{"ms":118290072}}, "memstats":{"gc_next":52
521504, "memory_alloc":27487040, "memory_total":3571799880}, "runtime":{"goroutines":493}, "filebeat":{"
harvester":{"open_files":3, "running":2}}, "libbeat":{"config":{"module":{"running":0}}, "output":{"we
ad":{"errors":1}, "write":{"bytes":125}}, "pipeline":{"clients":4, "events":{"active":4118, "retry":3}},
"registrar":{"states":{"current":4}}, "system":{"load":{"1":0.13, "15":0.01, "5":0.03, "nonm":{"1":0.13
,"15":0.01, "5":0.03}}}}]}
```

Logging

# Définition du Monitoring

Le monitoring est la surveillance continue d'un système ou d'une application pour détecter les anomalies et les problèmes potentiels.



# Définition du Logging

Le logging est l'enregistrement de données sur l'état et les activités d'un système.

Il permet de récupérer des informations sur les erreurs et les anomalies qui se produisent, et de les utiliser pour résoudre les problèmes, comprendre les tendances et identifier les tendances à venir.

Les données de journalisation peuvent également être utilisées pour répondre aux exigences réglementaires ou pour la conformité.

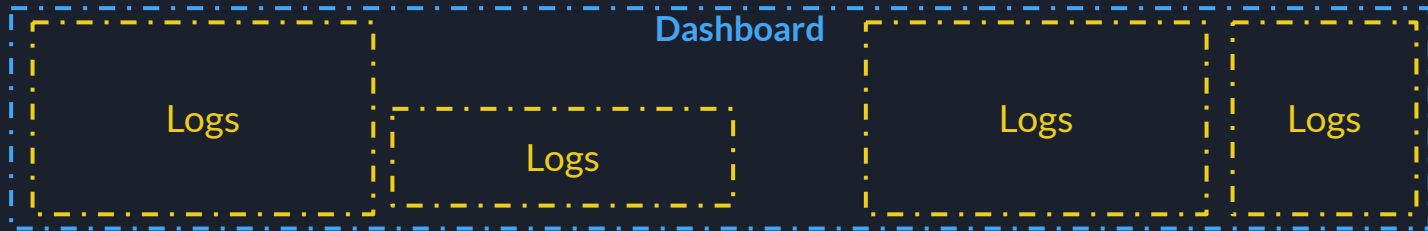
```
Nov 29 10:10:02 lograzer filebeat[21206]: 2020-11-29T10:10:02.2782Z#011INFO#011[monitoring]#011log/lo
g.go:145#011Non-zero metrics in the last 30s#011{"monitoring": {"metrics": {"beat":{"cpu":{"system":
{"ticks":167030},"total":{"ticks":360860,"time":{"ms":31,"value":360860},"user":{"ticks":193830,"tim
e":{"ms":31}}},"handles":{"limit":{"hard":4096,"soft":1024},"open":12},"info":{"ephemeral_id":"ec16b3
47-4538-416b-a781-5892dd598d34"},"uptime":{"ms":1182960072},"memstats":{"gc_next":52521504,"memory_a
lloc":27085744,"memory_total":35717398272},"runtime":{"goroutines":49},"filebeat":{"harvester":{"op
en_files":3,"running":2},"libbeat":{"config":{"module":{"running":0},"pipeline":{"clients":4,"even
ts":{"active":4118}}},"registrar":{"states":{"current":4},"system":{"load":{"1":0,"15":0,"5":0,"nor
m":{"1":0,"15":0,"5":0}}}}}}}}
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1222Z#011ERROR#011pipeline/output.go:10
0#011Failed to connect to backoff(elasticsearch(http://192.168.1.114:9200)): Get http://192.168.1.11
4:9200: EOF
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1222Z#011INFO#011pipeline/output.go:93#
011Attempting to reconnect to backoff(elasticsearch(http://192.168.1.114:9200)) with 1570 reconnect
attempt(s)
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232Z#011INFO#011[publisher]#011pipelin
e/retry.go:196#011retryer: send unwait-signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232Z#011INFO#011[publisher]#011pipelin
e/retry.go:198#011 done
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232Z#011INFO#011[publisher]#011pipelin
e/retry.go:173#011retryer: send wait signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232Z#011INFO#011[publisher]#011pipelin
e/retry.go:175#011 done
Nov 29 10:10:32 lograzer filebeat[21206]: 2020-11-29T10:10:32.2792Z#011INFO#011[monitoring]#011log/lo
g.go:145#011Non-zero metrics in the last 30s#011{"monitoring": {"metrics": {"beat":{"cpu":{"system":
{"ticks":167030,"time":{"ms":31},"total":{"ticks":360870,"time":{"ms":61,"value":360870},"user":{"ti
cks":193840,"time":{"ms":31}}},"handles":{"limit":{"hard":4096,"soft":1024},"open":12},"info":{"ephem
eral_id":"ec16b347-4538-416b-a781-5892dd598d34"},"uptime":{"ms":1182990072},"memstats":{"gc_next":52
521504,"memory_alloc":27487040,"memory_total":3571779558},"runtime":{"goroutines":49},"filebeat":{"
harvester":{"open_files":3,"running":2},"libbeat":{"config":{"module":{"running":0},"output":{"re
ad":{"errors":1},"write":{"bytes":125},"pipeline":{"clients":4,"events":{"active":4118,"retry":3}}}}
},"registrar":{"states":{"current":4},"system":{"load":{"1":0.13,"15":0.01,"5":0.03,"norm":{"1":0.13
,"15":0.01,"5":0.03}}}}}}}}}
```

Logging

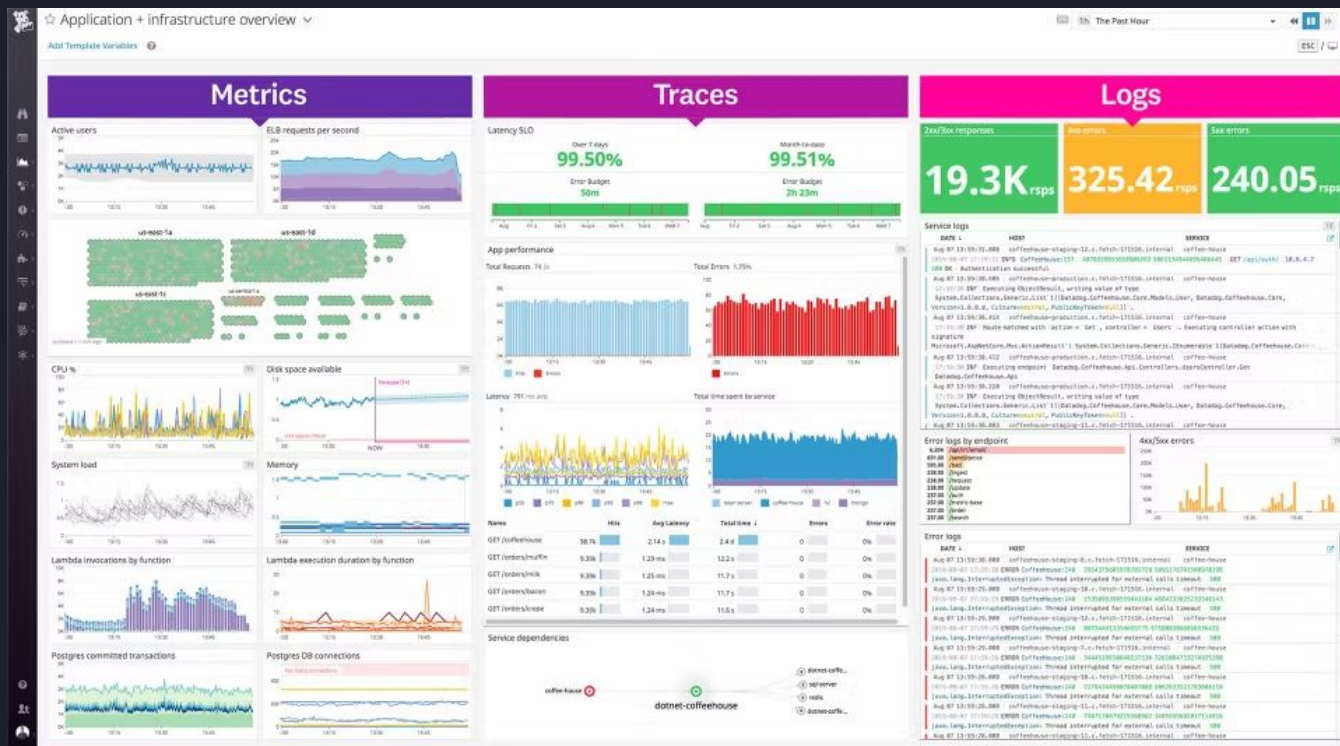


# Le monitoring

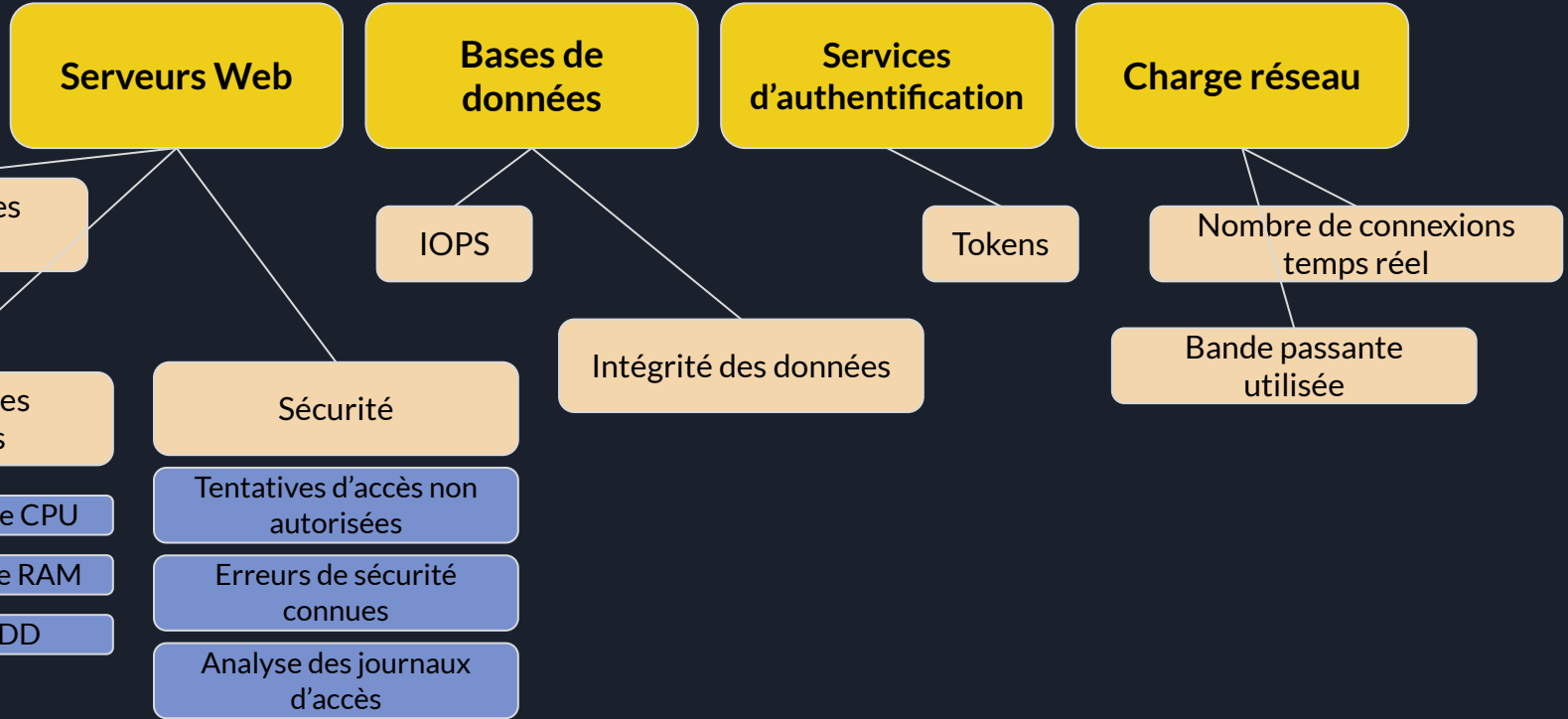
Les technologies de monitoring comme Datadog permettent aux administrateurs de systèmes de surveiller les performances de leurs services web en temps réel, collecter des métriques et des données de journalisation de leurs applications, et utiliser des alertes et des rapports pour identifier les problèmes potentiels avant qu'ils ne causent des perturbations de service.



# Exemple d'outil de Monitoring

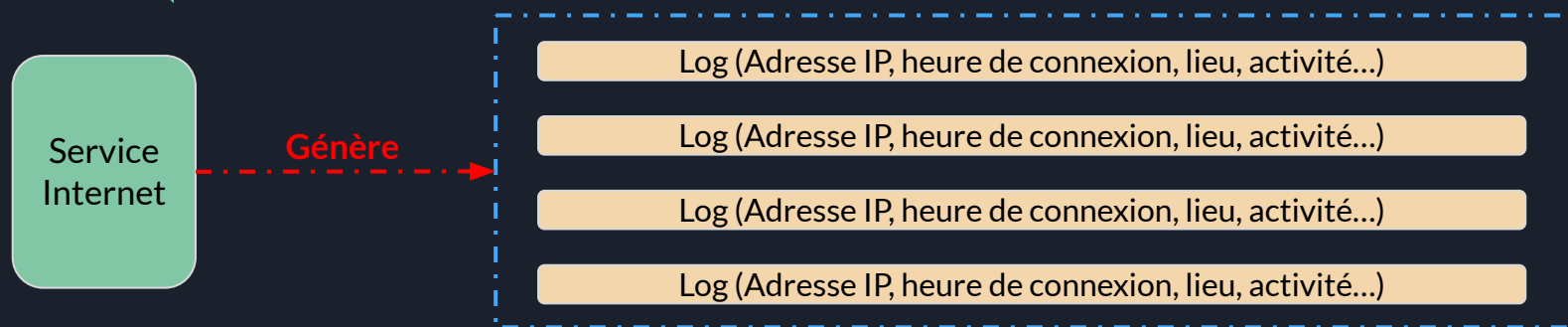


# Les objets à monitorer





# Objectif du logging



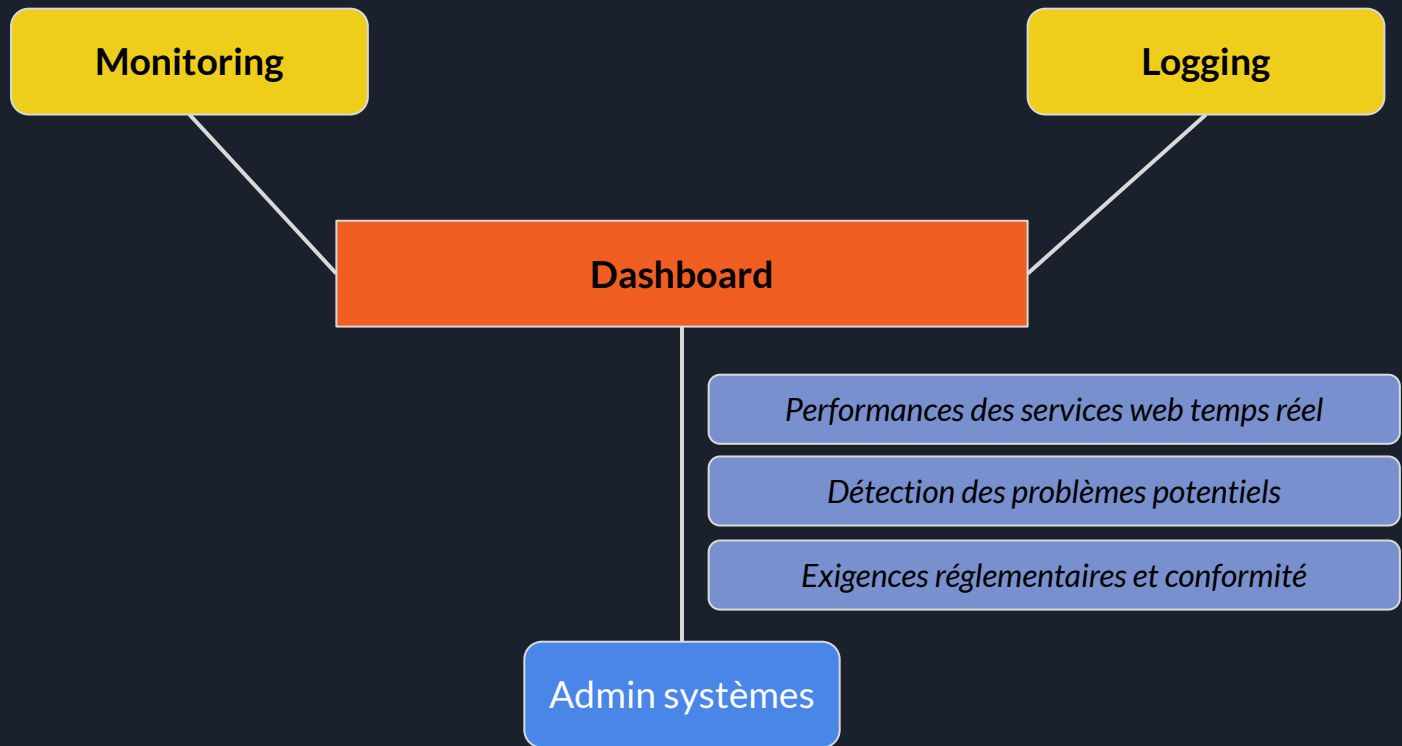
## Analyse Préventive

Détecter un comportement inhabituel sur un SI et remonter le comportement pour voir si c'est un faux positif ou un évènement de cybersécurité

## Forensic

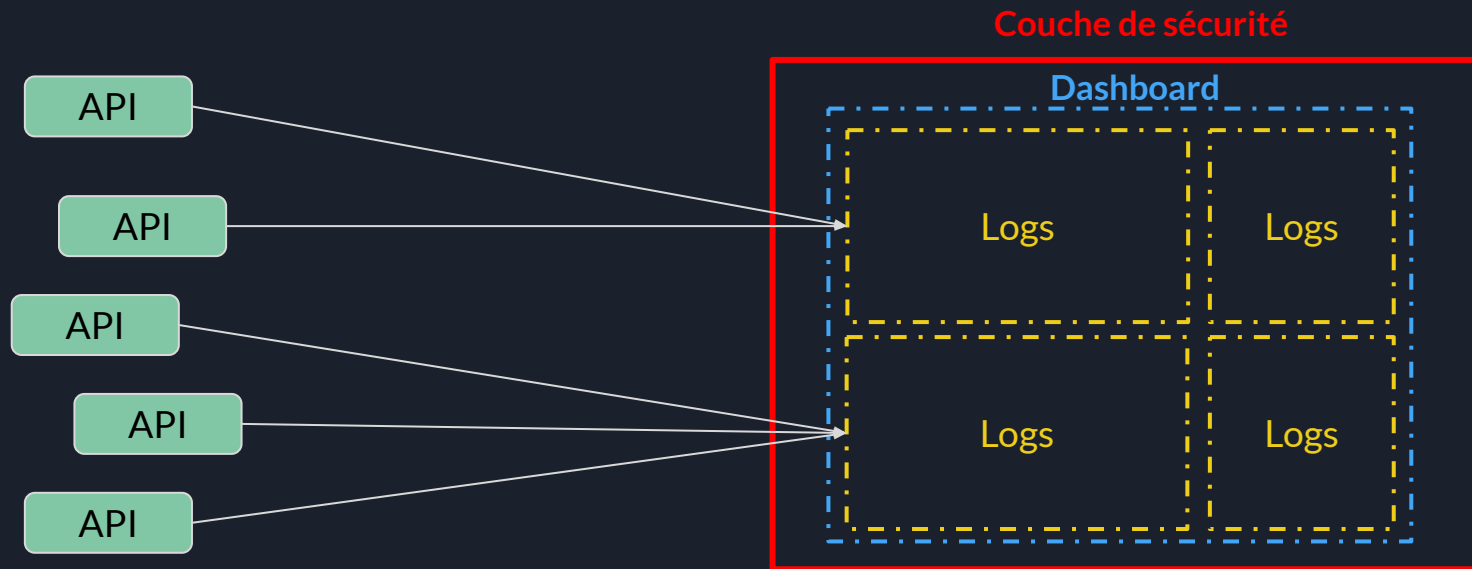
“Police scientifique” Étude des logs pour trouver des traces des pirates qui ont organisé une attaque

# Dualité monitoring/logging



# Sécurité des outils de monitoring

Il est important de noter que la sécurité de ces outils de monitoring et de logging doit être prise en considération pour éviter les fuites de données ou les accès non autorisés aux données collectées. Il est donc essentiel de mettre en place les protocoles de sécurité adéquats et de suivre régulièrement les politiques de sécurité.





# Logging et monitoring en python

Python dispose d'une bibliothèque de journalisation intégrée appelée **logging**, qui permet de générer des messages de journalisation de manière simple et efficace. Il est également possible d'utiliser des bibliothèques de monitoring telles que psutil pour récupérer des informations sur les performances et les ressources système.

## logging (bibliothèque)

basicConfig

info

warning

error

## psutil (bibliothèque)

virtual\_memory

total

used

available

## psutil (bibliothèque)

cpu\_percent

# Journalisation de base

psutil (bibliothèque)

basicConfig

info

warning

error

```
import logging

logging.basicConfig(level=logging.INFO,
                    format = '%(asctime)s %(levelname)s %(message)s')

logging.info("Application started")
logging.warning("A warning message")
logging.error("An error occurred")
```

# Monitoring de la mémoire utilisée

psutil (bibliothèque)

basicConfig

info

warning

error

```
import psutil

memory_info = psutil.virtual_memory()
print("Total memory : ", memory_info.total)
print("Used memory : ", memory_info.used)
print("Available memory : ", memory_info.available)
```



# Monitoring de l'utilisation du processeur

psutil (bibliothèque)

cpu\_percent

```
import psutil
```

```
cpu_info = psutil.cpu_percent(interval=1)  
print("CPU usage : ", cpu_info)
```



# Journalisation et monitoring combinés

```
import psutil
import logging

logging.basicConfig(level=logging.INFO,
                    format = '%(asctime)s %(levelname)s %(message)s')

logging.info("Application started")

#Perform calculations
result = 0
for i in range (1, 101):
    result += i
    cpu_info = psutil.cpu_percent(interval = 1)
    logging.info("CPU usage : %s", cpu_info)

logging.info("Application finished")
```





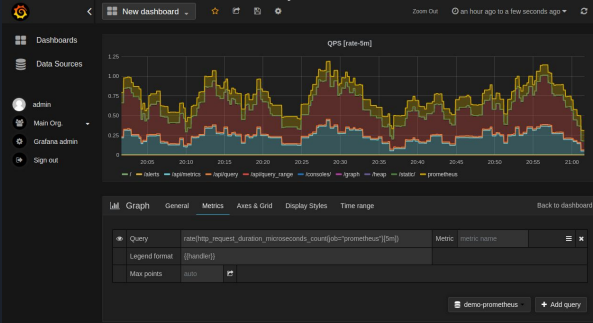
# Journalisation et monitoring combinés

Il est important de noter que les exemples ci-dessus ne représentent qu'une petite partie des fonctionnalités offertes par les bibliothèques de journalisation et de monitoring disponibles en Python. Il est donc important de consulter la documentation de ces bibliothèques pour en savoir plus sur les fonctionnalités.

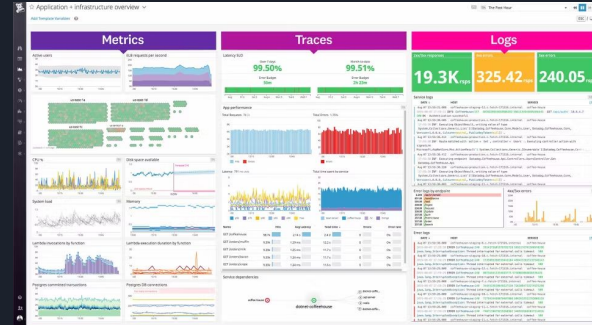
# Etude de cas : Les outils pour du monitoring et log

## Etude de cas (30 minutes)

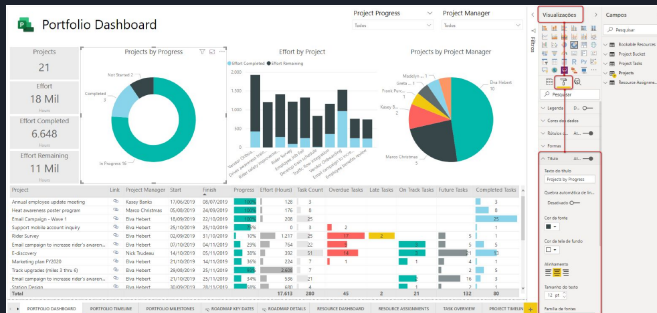
### Groupe 1 Grafana/Prometheus



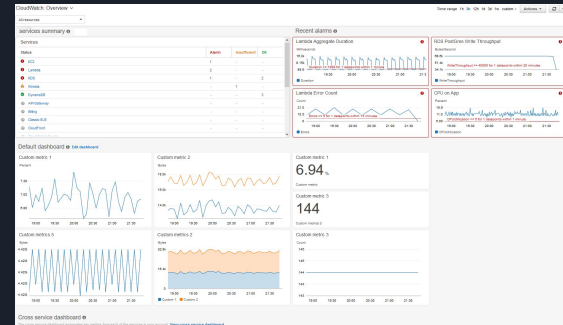
### Groupe 2 DataDog



### Groupe 3 PowerBI



### Groupe 4 AWS CloudWatch



# Grafana

Tableaux de bord interactifs

Dashboard personnalisé Grafana

Graphiques

Prometheus

InfluxDB

Graphite

Elasticsearch

Ingestion temps réel

Données de performance

Format de données



Open source

Visualisation de données

Monitoring

Analyse de tendances

Alertes temps réel

Collaboration temps réel

# Grafana

## Dashboard personnalisé Grafana



### Équipe DevOps

Surveillance de l'infrastructure temps réel

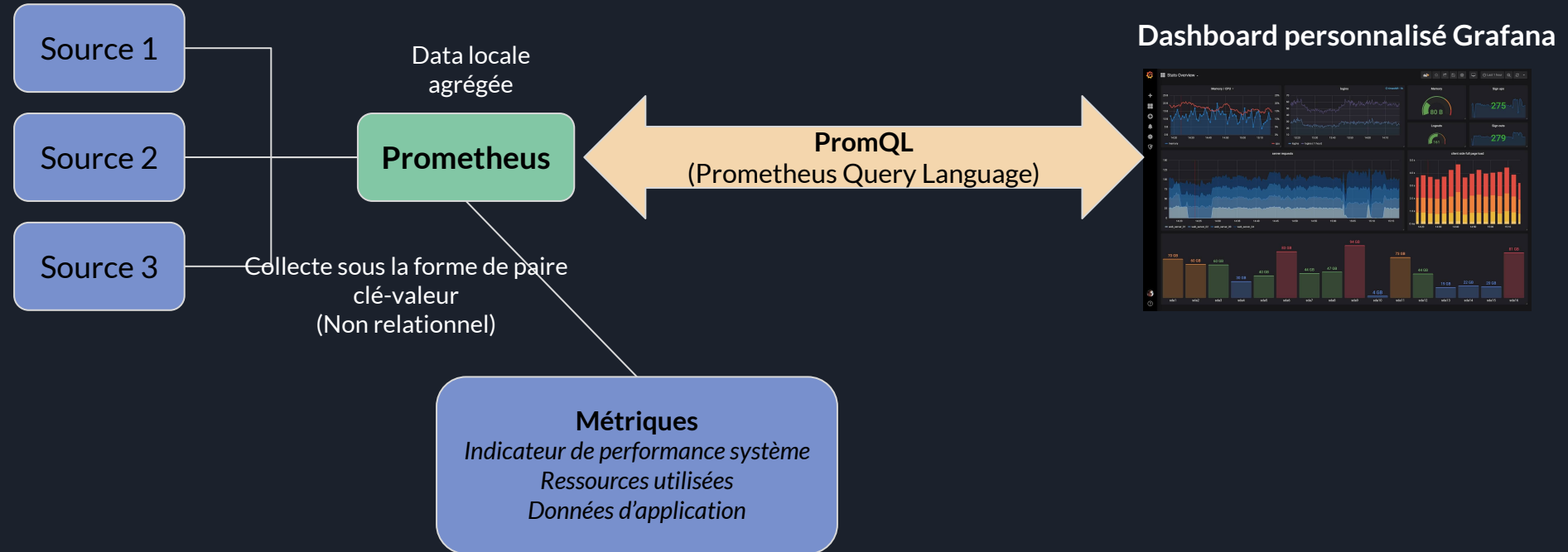
### Équipe Cybersécurité

Identification des problèmes de sécurité temps réel

### Équipe Gestion de performance

Gestion des performances temps réel

# Grafana et Prometheus



# Grafana et Prometheus

Ingestion de données (temps réel)

Data Visualization

Source 1

Source 2

Source 3

Data locale  
agrégée

**Prometheus**

Collecte sous la forme de paire  
clé-valeur  
(Non relationnel)

**Métriques**

*Indicateur de performance système  
Ressources utilisées  
Données d'application*

**PromQL**  
(Prometheus Query Language)

**Dashboard personnalisé Grafana**



# Travaux pratiques : Création d'un dashboard Grafana avec une source de donnée Prometheus

**Objectif du TP : Réaliser le Lab sur Grafana**

<https://grafana.com/tutorials/grafana-fundamentals/>

Ajouter des sources de données

Créer un Dashboard

Réaliser un fichier pdf avec captures d'écran expliquant vos résultats.

Deadline de rendu : **Mercredi 21 février 23h59**

Format attendu : "TP\_Grafana\_NOM\_PRENOM"

Adresse de rendu : yann.fornier@gmail.com



# Datadog

Datadog est une **plateforme de surveillance** complète

Collecte de métriques

Analyse de logs

Tracing d'applications

Alerting

Intégration avec des services Cloud



Grafana est davantage une plateforme de visualisation de données



# Datadog

Services Web

API

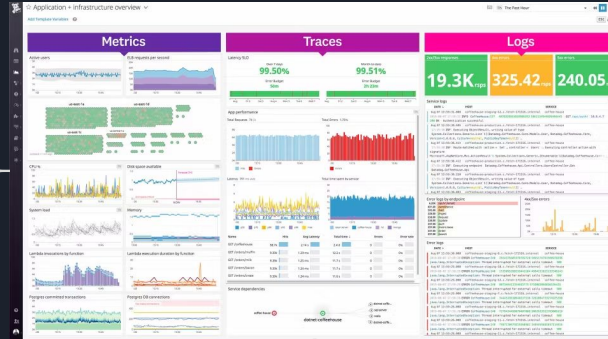
API

API

API

API

Métriques stockées dans un système distribué  
Indicateur de performance système  
Ressources utilisées  
Données d'application



Cloud

AWS

Azure

Google Cloud Platform

Alertes pour des valeurs limites

Utilisation de ressources anormales

Erreurs

Analyse de journaux

Surveillance de l'infrastructure

Performance des requêtes

# Travaux pratiques : Création d'un APM (Application Performance Monitoring) Datadog

**Objectif du TP : Réaliser le Lab sur Datadog**

<https://learn.datadoghq.com/courses/dd-101-dev>

Réaliser un fichier pdf avec captures d'écran expliquant vos résultats.

Deadline de rendu : **Mercredi 21 février 23h59**

Format attendu : "TP\_DATADOG\_NOM\_PRENOM"

Adresse de rendu : [yann.fornier@gmail.com](mailto:yann.fornier@gmail.com)



# PowerBI

PowerBI est une suite d'outils d'analyse commerciale développée par Microsoft. Elle permet de faire de la visualisation et de l'analyse de donnée à partir de différentes sources.



*Un dashboard PowerBI*

# PowerBI

Modélisation de  
données

PowerBI

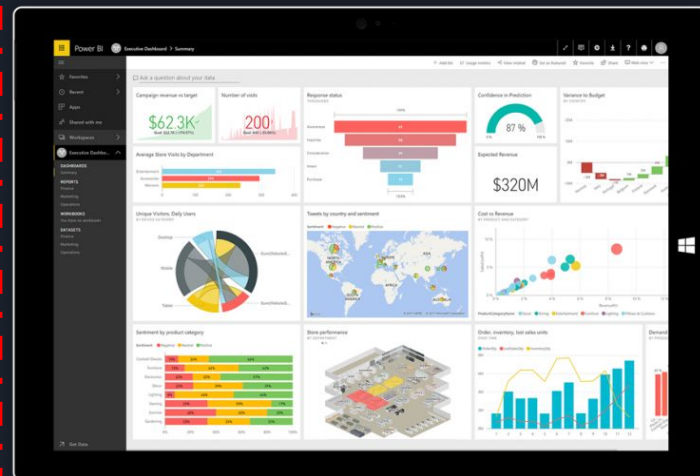
Données  
diverses

Excel

Azure

Dynamics  
365

Services  
Microsoft



Création de  
visualisations

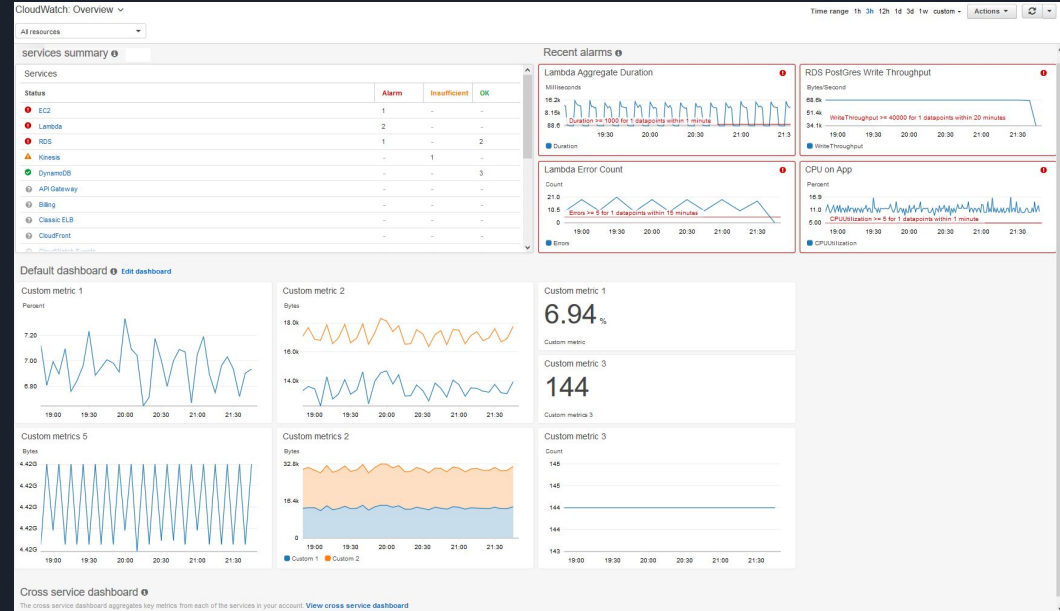
Analyse avancée

Génération de  
rapports

Modèles statistiques

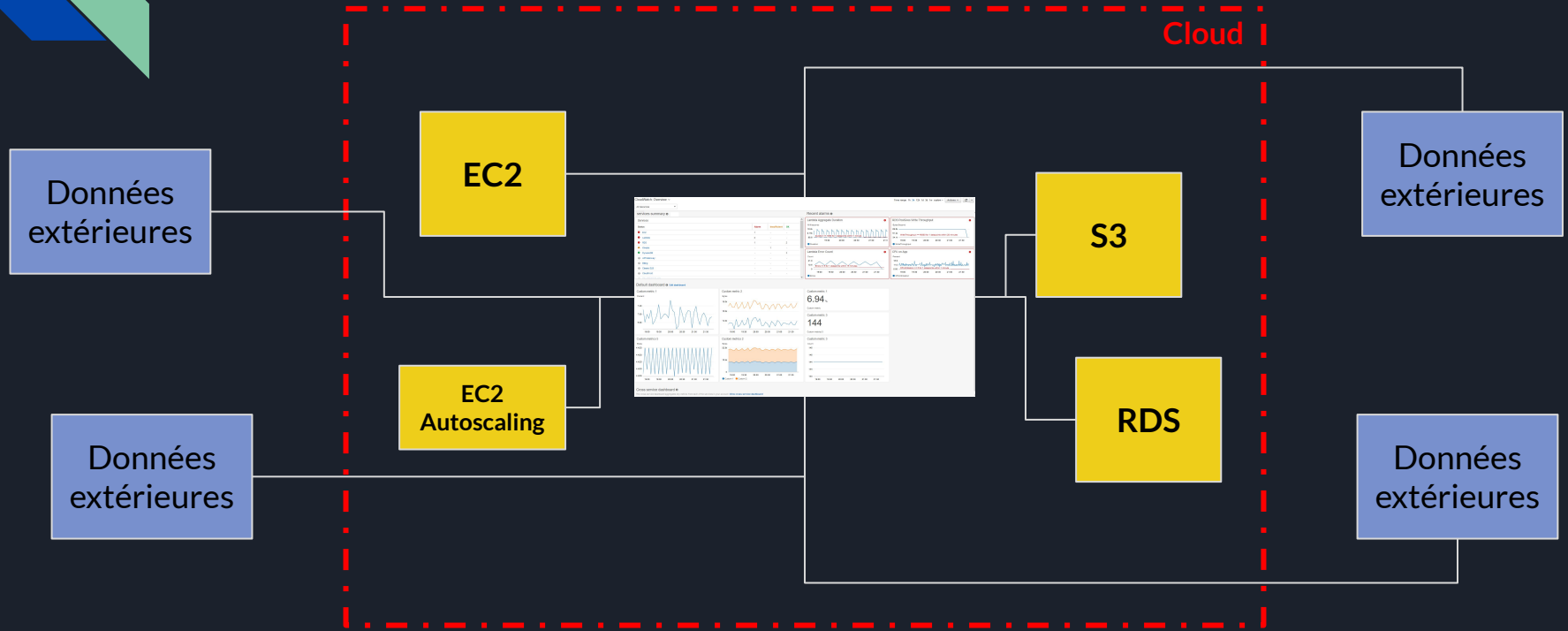
# AWS Cloudwatch

AWS Cloudwatch est la plateforme de surveillance et de gestion des performances d'Amazon Web Services



*Un dashboard CloudWatch*

# AWS CloudWatch



# AWS CloudWatch

