ENSIAS

جامعة محمد الخامس بالرباط
Université Mohammed V de Rabat

SUPERVISION TEMPS RÉEL DES LOGS FIREFOX AVEC LA PILE ELK + MACHINE LEARNING

RÉALISÉ PAR:
BOUTANFIT SALMA
JENNANE SALMA
EL IDRISSI ASMA

ENCADRÉ PAR:
M.KERZAZI NOUREDDINE

Sommaire

Contexte général

Les systèmes de build comme Firefox génèrent un grand volume de logs difficiles à analyser manuellement.

Le projet vise à mettre en place une solution de supervision temps réel basée sur la stack ELK (Elasticsearch, Logstash, Kibana) pour centraliser, analyser et visualiser ces logs.

Un module de Machine Learning permet en plus de détecter automatiquement les anomalies dans les flux de données.

Volume élevé

Les systèmes contemporains génèrent une quantité considérable de journaux.

Détection manuelle

Cela complique la détection manuelle des anomalies.

Problématique

- Volume de logs très important généré par Firefox
- Analyse manuelle lente et difficile
- Besoin d’une solution automatique et en temps réel

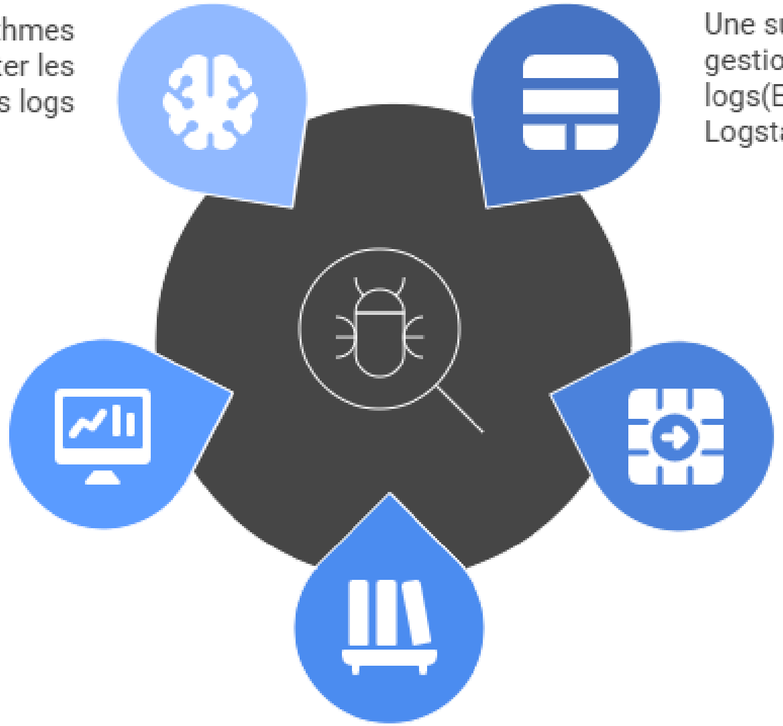
Objectifs

Intégration du Machine Learning

Utiliser des algorithmes pour détecter les anomalies dans les logs

Stack ELK

Une suite d'outils pour la gestion et l'analyse des logs(Elasticsearch, Logstash, Kibana)



Dashboard de supervision

Une interface visuelle pour surveiller les données en temps réel

Centralisation des logs

Consolider les logs de diverses sources en un seul emplacement

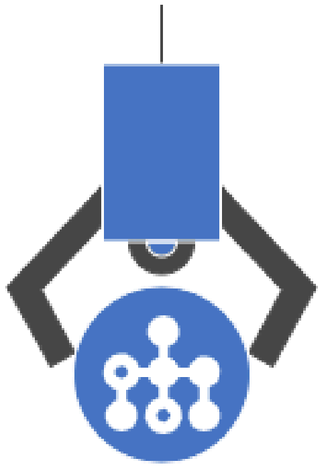
Structuration des logs

Organiser les logs dans un format cohérent et lisible

Données de build Firefox

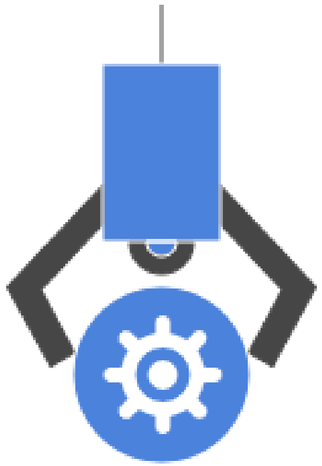
```
04:11:58  WARNING -  INFO | Failed: 1
04:11:58  WARNING -  One or more unittests failed.
04:11:58     INFO -  INFO | Todo: 4
04:11:58     INFO -  INFO | Retried: 5
04:11:58     INFO -  SUITE-END | took 1184s
04:11:59    ERROR - Return code: 1
04:11:59     INFO - TinderboxPrint: xpcshell-xpcshell<br/>3218/<em class="testfail">1</em>
04:11:59  WARNING - # TBPL FAILURE #
04:11:59  WARNING - setting return code to 2
04:11:59  WARNING - The xpcshell suite: xpcshell ran with return status: FAILURE
04:11:59     INFO - Running post-action listener: _package_coverage_data
04:11:59     INFO - Running post-action listener: _resource_record_post_action
04:11:59     INFO - [mozharness: 2018-06-08 11:11:59.141993Z] Finished run-tests step (success)
04:11:59     INFO - Running post-run listener: _resource_record_post_run
04:11:59  WARNING - error reading instance_metadata: Traceback (most recent call last):
04:11:59  WARNING -   File "/builds/slave/test/scripts/mozharness/base/python.py", line 456, in perfherder_resource_options
04:11:59  WARNING -     instance = im['aws_instance_type'].encode('ascii')
04:11:59  WARNING - KeyError: 'aws_instance_type'
04:11:59     INFO - Validating Perfherder data against /builds/slave/test/scripts/external_tools/performance-artifact-schema.json
```

Composantes du système



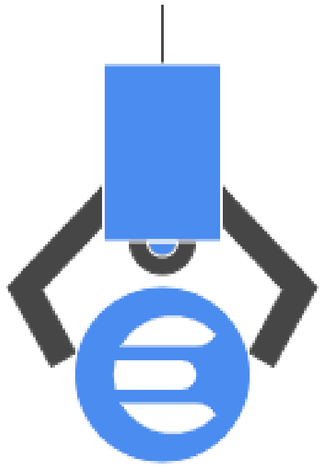
Filebeat

Collecte les fichiers de logs.



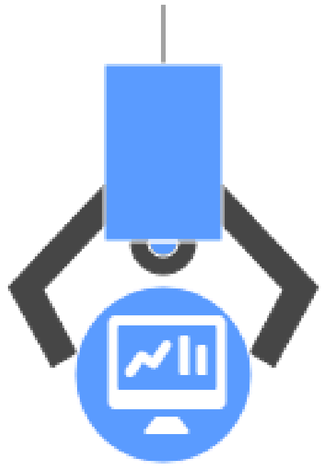
Logstash

Transforme et nettoie les données.



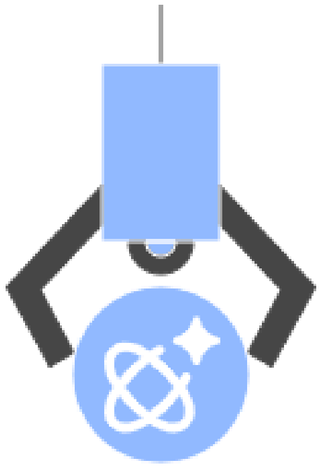
Elasticsearch

Indexe et rend les logs consultables.



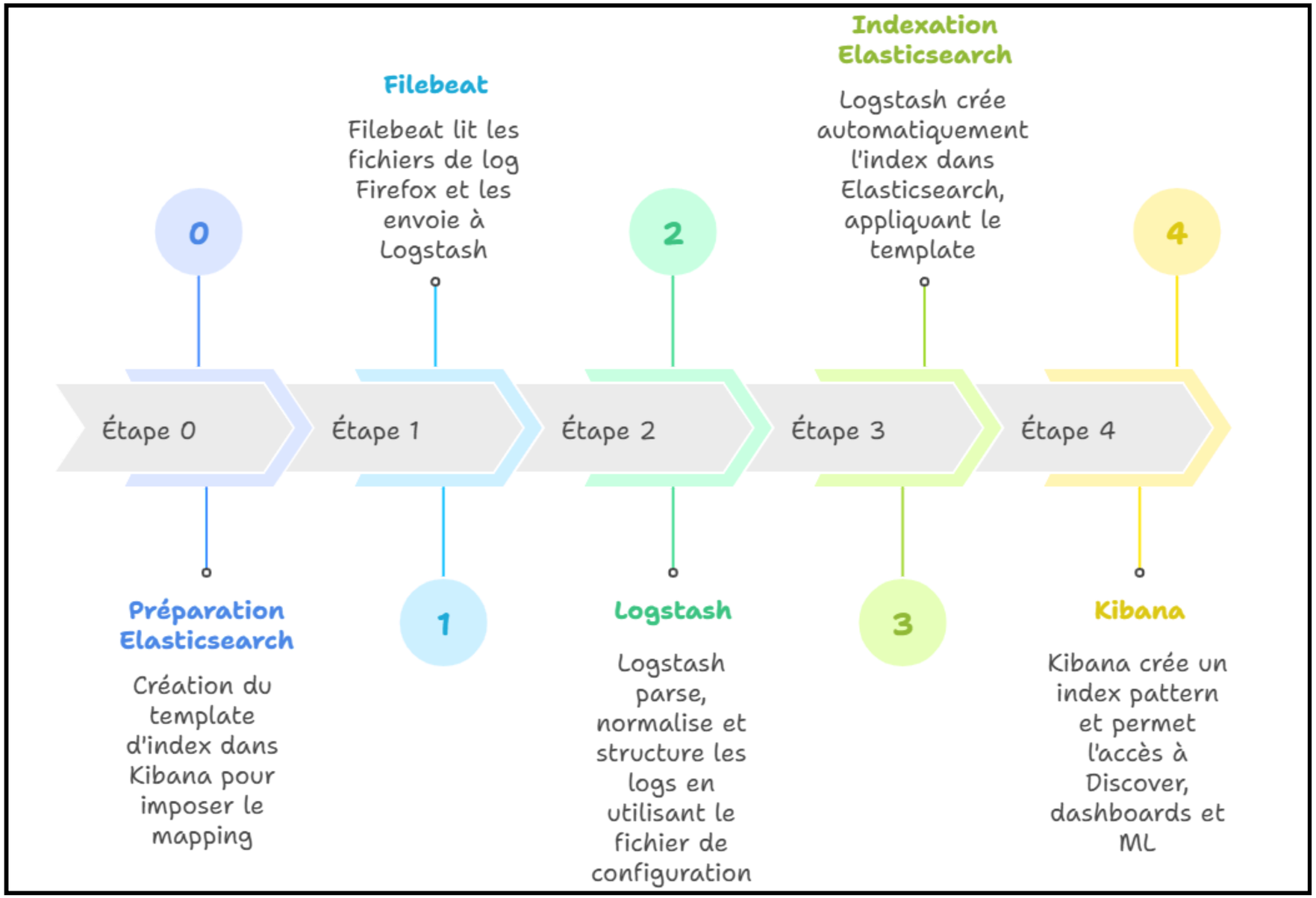
Kibana

Visualise les données en temps



Module Machine Learning

Étapes du Pipeline de Log Firefox



Collecte des Logs (Filebeat)

Filebeat surveille les fichiers de logs générés par Firefox et les envoie vers Logstash. Il est léger et conçu pour transmettre un flux constant de logs sans perte.

Vue d'ensemble du pipeline Logstash

```
input { beats { port => 5044 } }

filter {
  # (1) extraire date
  # (2) grok multi-formats
  # (3) @timestamp
  # (4) normalisation & conversions
  # (5) métadonnées
  # (6) tags & enrichissement
  # (7) drop bruit
  # (8) ECS fields
}

output {
  elasticsearch { index => "firefox-logs-%{+YYYY.MM.dd}" }
  stdout { codec => rubydebug }
}
```

Transformation et Nettoyage (Logstash)

Logstash est le moteur ETL de la solution. Il parse les lignes de logs, en extrait la date, l'heure et le niveau d'erreur, puis reconstruit des documents structurés avant de les envoyer à Elasticsearch.

Indexation et Stockage (Elasticsearch)

- Elasticsearch stocke les logs sous forme de documents JSON indexés, ce qui permet d'effectuer des recherches rapides et des analyses statistiques en temps réel.

Problèmes d'indexation par défaut

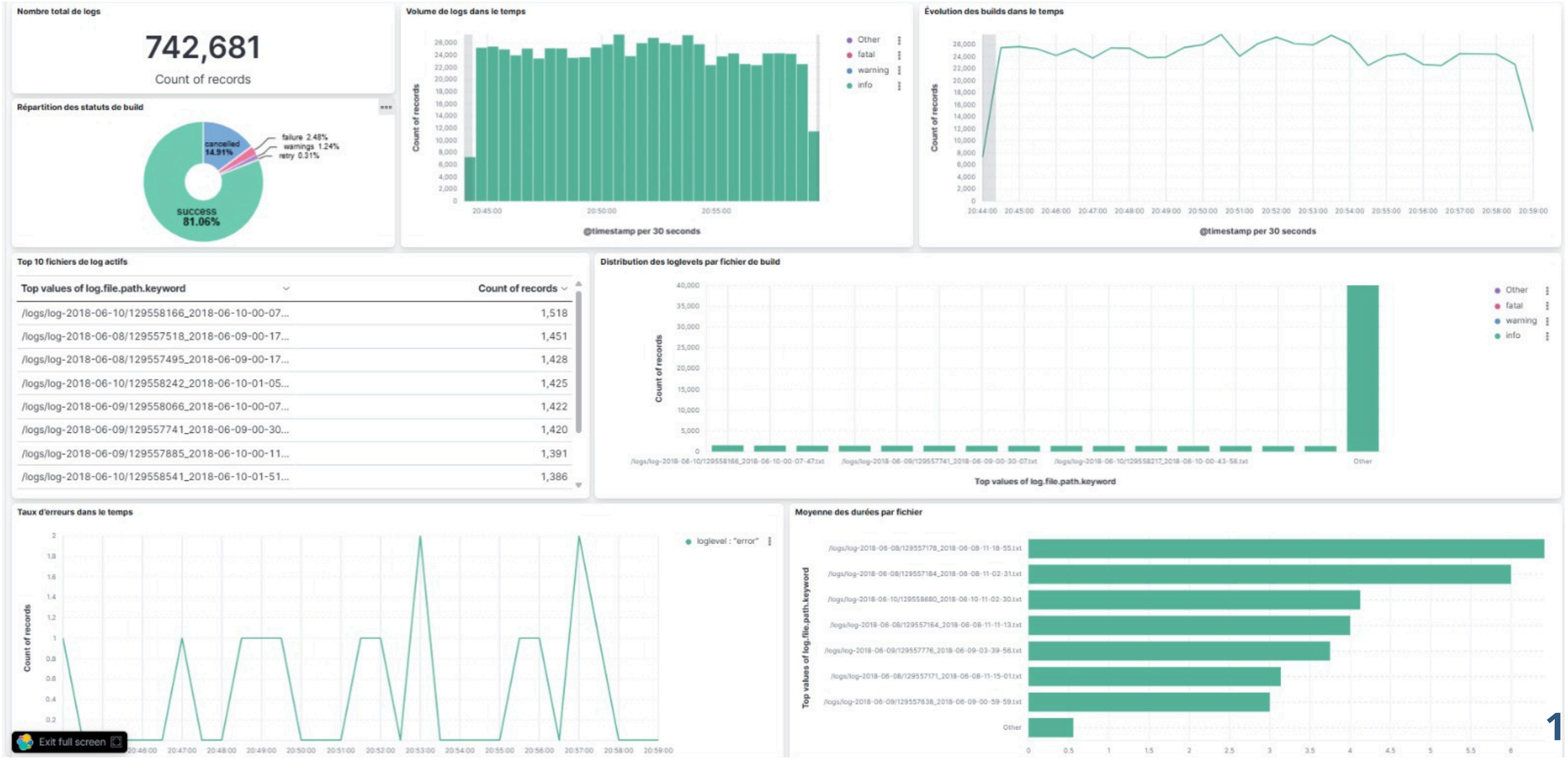
- Mauvaise détection des types
- Types incohérents
- Création de champs inutiles

Mapping statique et typage strict

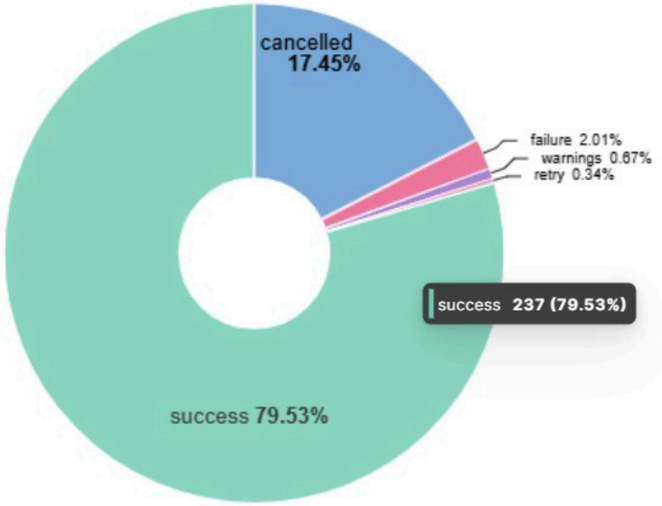
History Settings Help

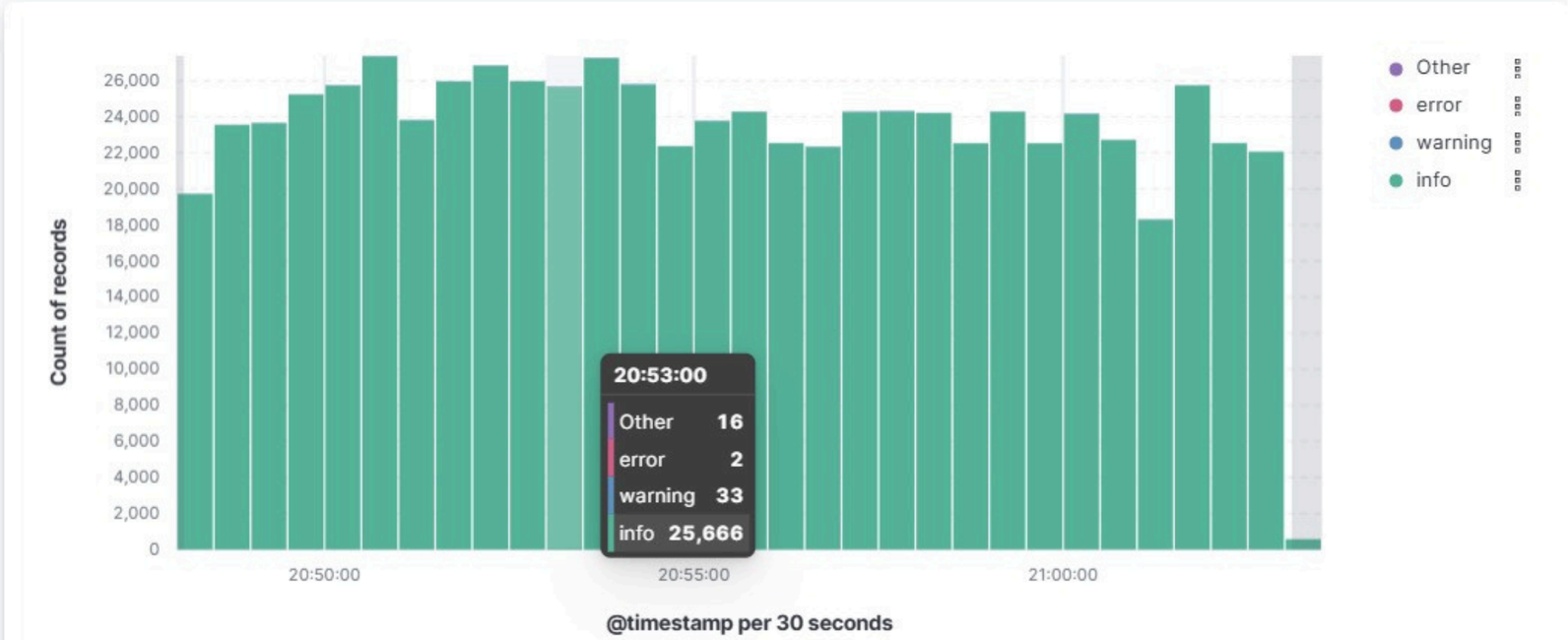
```
PUT _index_template/firefox_build_logs_template
{
  "index_patterns": ["firefox-logs-*"],
  "template": {
    "settings": {
      "index.lifecycle.name": "firefox_logs_policy",
      "index.lifecycle.rollover_alias": "firefox-logs",
      "index.number_of_shards": 1
    },
    "mappings": {
      "properties": {
        "@timestamp": { "type": "date" },
        "log.file.path": { "type": "keyword" },
        "log_type": { "type": "keyword" },

        "loglevel": { "type": "keyword" },
        "message": { "type": "text" },

        "field_name": { "type": "keyword" },
        "field_value": { "type": "keyword" },

        "results_text": { "type": "keyword" },
        "results_count": { "type": "integer" },

        "step_status": { "type": "keyword" },
        "step_name": { "type": "text" },
        "results_code": { "type": "integer" },
        "step_elapsed_secs": { "type": "float" },

        "step_at_ts": {
          "type": "date",
          "format": "yyyy-MM-dd HH:mm:ss.SSSSSS||yyyy-MM-dd HH:mm:ss.SSS||yyyy-MM-dd HH:mm
            :ss||strict_date_optional_time||epoch_millis",
          "ignore_malformed": true
        },

        "exit_code": { "type": "integer" },

        "line_time": { "type": "keyword" },
        "transfer_rate": { "type": "float" },
        "transfer_unit": { "type": "keyword" },
        "downloaded_file": { "type": "keyword" },
        "bytes_downloaded": { "type": "long" },
        "total_bytes": { "type": "long" }
      }
    }
```

Visualisation et Analyse (Kibana)

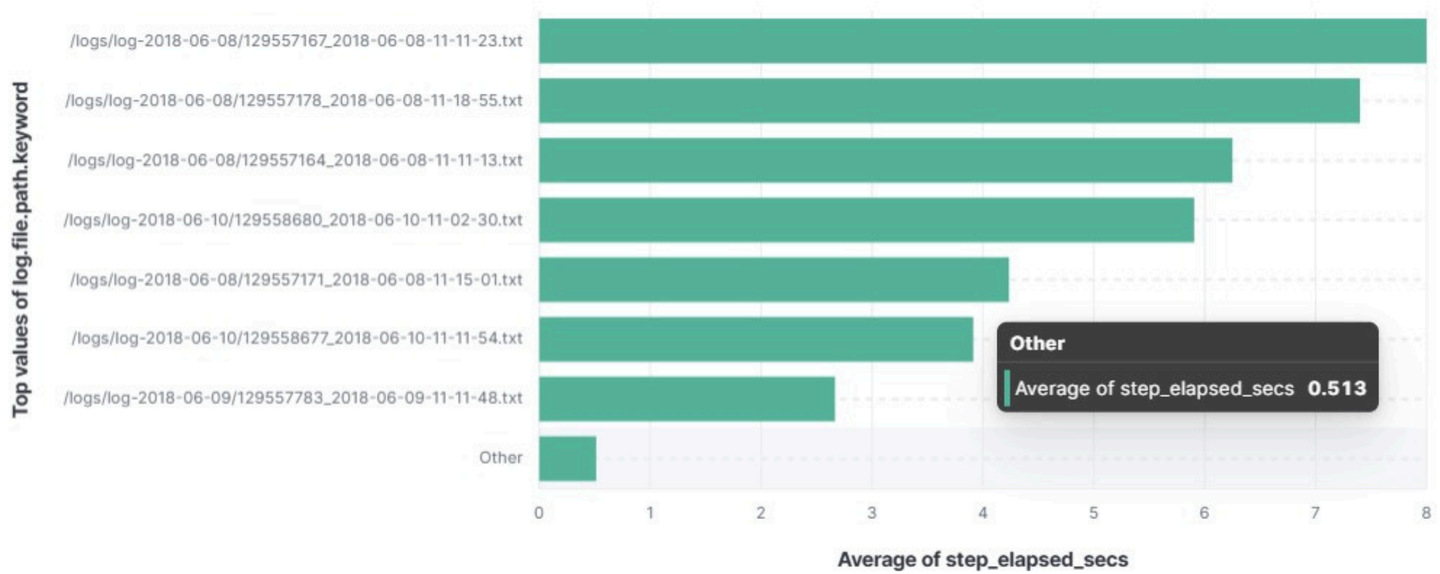


Visualisation et Analyse (Kibana)

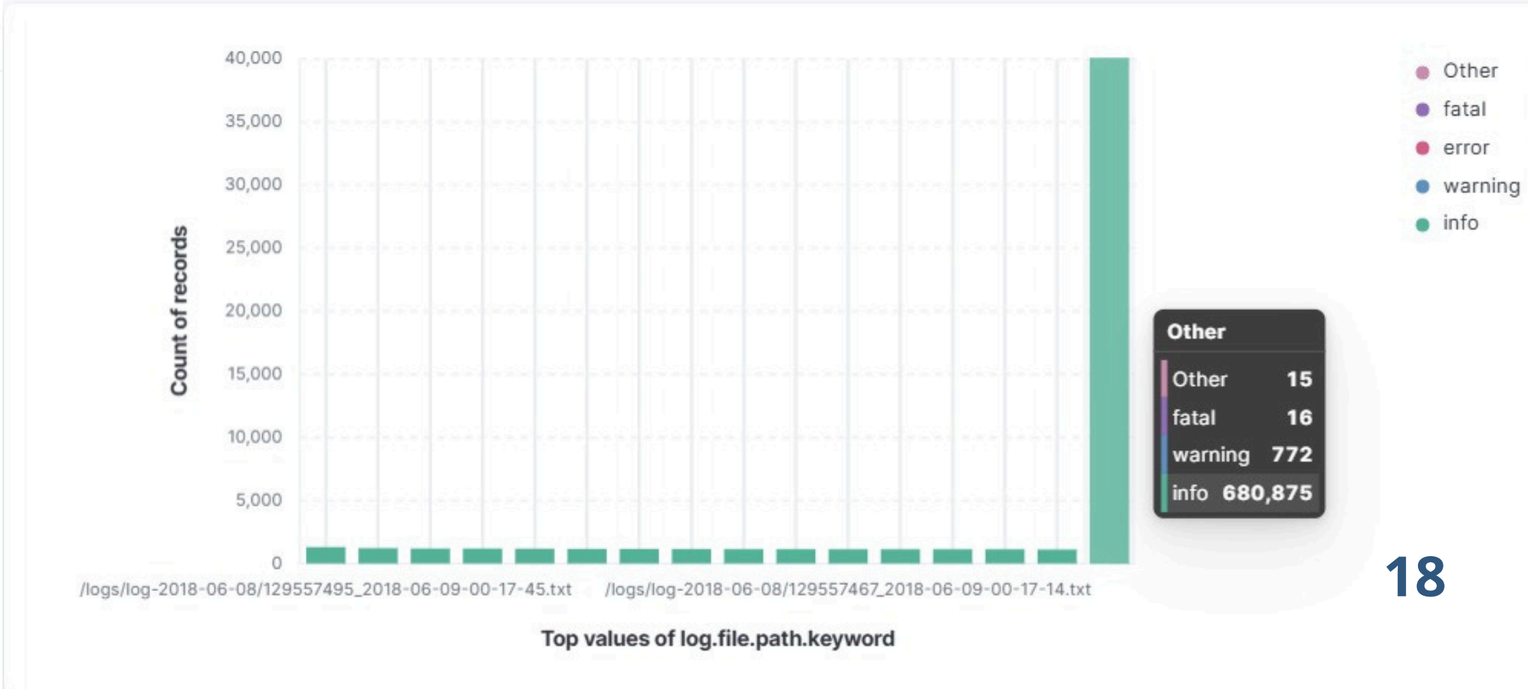




Visualisation et Analyse (Kibana)



Visualisation et Analyse (Kibana)



Détection d'Anomalies (Machine Learning).

Détection d'Anomalies

Problématique

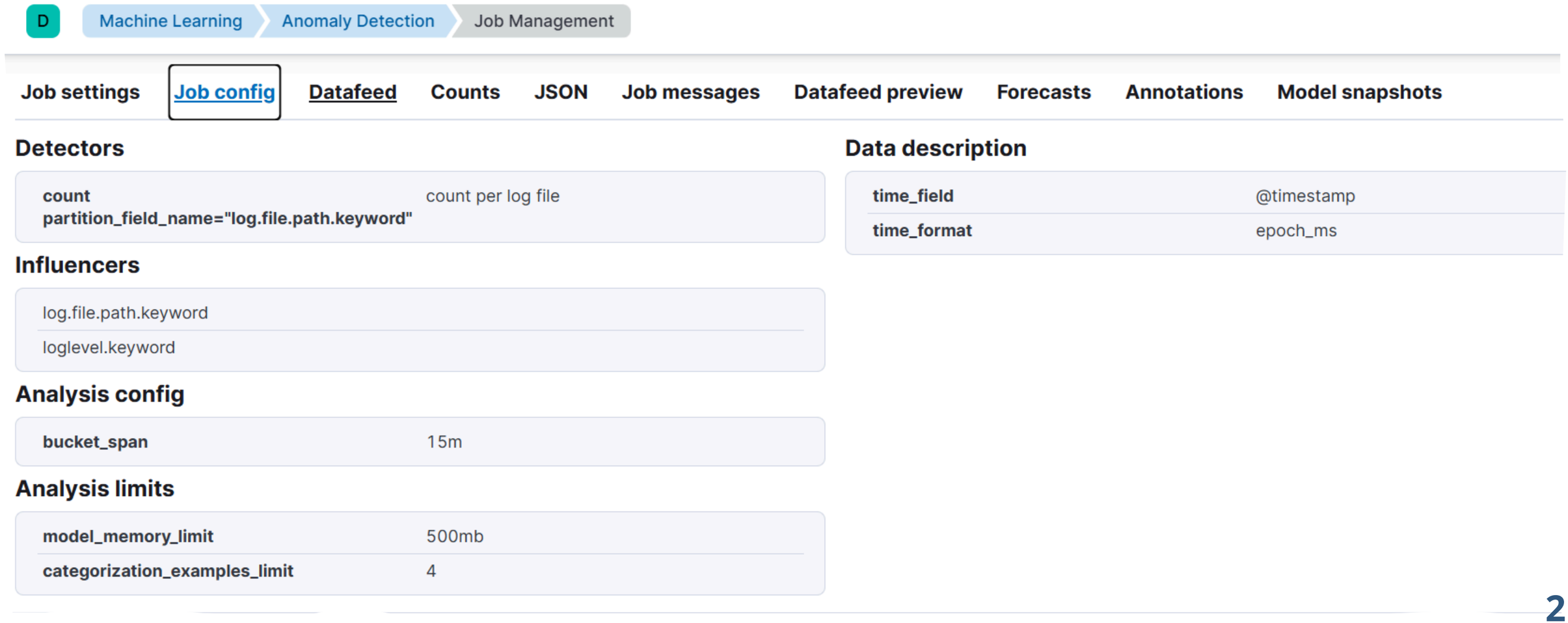
- Volume très élevé de logs (millions de lignes)
- Analyse manuelle = difficile / lente / erreurs possibles
- Besoin de surveillance proactive

Méthode utilisée

- Machine Learning non supervisé
- Le modèle apprend le comportement normal
- Détection automatique des écarts anormaux
- Sévérité notée de 0 à 100

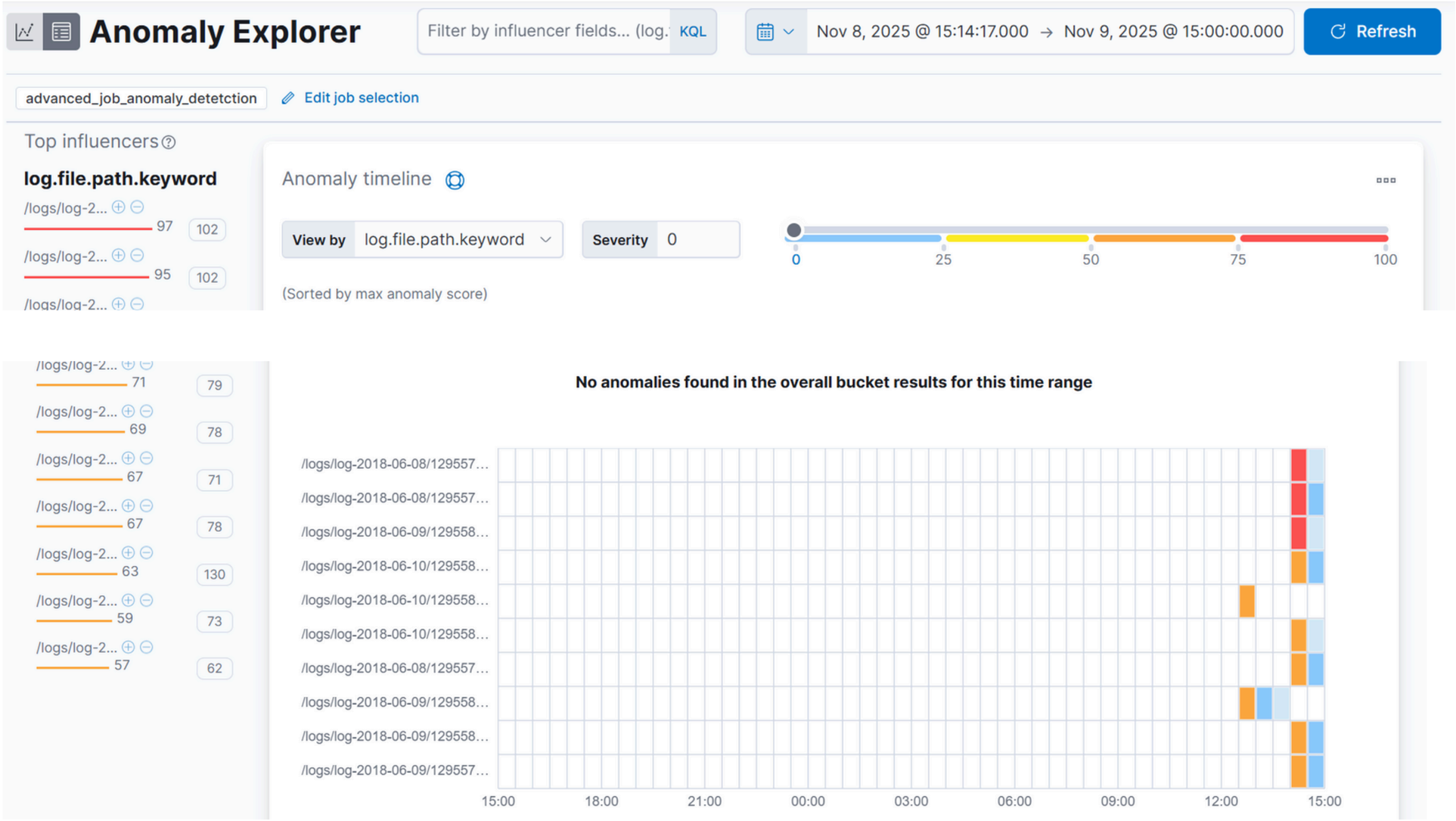
Détection d'Anomalies

Configuration du Job



Détection d'Anomalies

Résultats : Vue Globale



Détection d'Anomalies

Exemple d'Anomalie Identifiée

Time	Severity	Detector	Found for	Influenced by	Actual	Typical	Description	Actions
November 9th 2025, 14:00	99	count per log file	/logs/log-2018-06-08/129557474_2018-06-09-00-17-30.txt	log.file.path.keyword: /logs/log-2018-06-08/129557474_2018-06-09-00-17-30.txt ⊕ ⊖ loglevel.keyword : info ⊕ ⊖	10099	11.8	More than 100x higher	
November 9th 2025, 14:00	98	count per log file	/logs/log-2018-06-08/129557121_2018-06-08-00-33-57.txt	log.file.path.keyword: /logs/log-2018-06-08/129557121_2018-06-08-00-33-57.txt ⊕ ⊖ loglevel.keyword : info ⊕ ⊖	4113	7.58	More than 100x higher	

Résultats et Bénéfices

Le tableau de bord permet une compréhension rapide des erreurs et performances des builds Firefox. Les anomalies sont détectées automatiquement, ce qui réduit le temps d'analyse et améliore la fiabilité du monitoring.

Conclusion et Perspectives

•La solution ELK fournit une supervision complète et intelligente des logs. À l'avenir, elle peut être enrichie avec des alertes automatiques, de l'analyse sémantique des messages, et une intégration dans un pipeline DevOps complet.

Merci pour votre attention !