MASTERE SPECIALISE BIG DATA GESTION ET ANALYSE DES DONNEES MASSIVES

INF 726 Securite Big Data

Mise en place et manipulation d'une stack ELK

ELK stack pour l'analyse des paquets de données réseaux

Télécom Paris

Hugo Michel

Année 2021 / 2022

Table des matières

1. Introduction....3
2. Environnement technique....4
3. La pile ELK....5
3.1 Les motivations de la pile ELK....5
3.2 ElasticSearch....6
3.3 Kibana....6
3.4 Logstash....6
3.5 Packetbeat....7
3.6 Architecture ELK....7
4. Mise en place de la stack ELK....8
4.1 Installations....8
4.2 Description du fichier PCAP....11
4.3 Workflow du pipeline de données....11
4.4 Création d'un index sur Kibana....11
5. Dashboard Kibana....12
5.1 Discover....12
5.2 Le dashboard Kibana....13
6. Conclusion....15
7. Annexe....16
8. Bibliographie....16

1. Introduction

Avec l'utilisation croissante des terminaux mobiles (ordinateurs, téléphones, tablettes,…) et des objets connectés, plus de 2,5 quintillions octets transitent chaque jour sur le réseau mondial. Cette tendance ne semble pas s'atténuer puisque l'on estime que les données générées pourraient doubler tous les 2 ans. Par conséquent, de plus en plus de données sont collectées et stockées faisant des technologies Big Data des solutions indispensables pour permettre aux entreprises, tous secteurs d'activité confondus de traiter cette masse énorme d'information. Le véritable enjeu est alors de protéger toute cette masse de données. De nos jours, il s'avère que de nombreuses entreprises optent pour une stratégie axée sur la donnée. Ainsi ces dernières utilisent l'analyse Big data pour identifier les opportunités commerciales, améliorer les performances et orienter la prise de décision. Or de nombreux outils Big Data sont open source et ne sont pas conçus avec comme priorité numéro une la sécurisation des données. C'est pourquoi, à l'heure de la transformation numérique, les attaques et vols de données personnelles deviennent de plus en plus fréquent. Ces attaques ne sont pas sans conséquence pour les entreprises car celles-ci peuvent entraîner de graves répercussions tel que des pertes financières, des frais de justice et des amendes ou des sanctions.

Si la donnée est partout, la sécurité doit l'être tout autant. Le big data peut être utilisé pour analyser en continue le flux réseaux d'une entreprise et ainsi détecter des situations anormales sur le réseau et réagir le plus tôt possible à certaines attaques informatiques. Or dans un monde concurrentiel, les entreprises ne peuvent pas se permettre d'indisponibilité ou de ralentissement des performances de leurs applications car cela peut dégrader son image de marque.

C'est dans ce contexte que s'inscrit le projet ELK qui est dédié à la mise en place d'une pile logicielle pour l'analyse de grands ensembles de données réseau. Cette pile logicielle fait intervenir 3 technologies qui sont les suivantes :

- ElasticSearch
- Logstash
- Kibana

Figure 1 : Technologies ELK

Source : https://www.missioncloud.com/blog/what-is-elk-stack

La pile ELK est un ensemble de technologies open source pour la collecte, la recherche, l'analyse et la visualisation des données générées à travers diverses sources de données structurées et non structurées.

De nombreuse entreprises ont recourt à la pile ELK pour surveiller les performances, la disponibilité et la sécurité des applications issus du système d'information.

2. Environnement technique

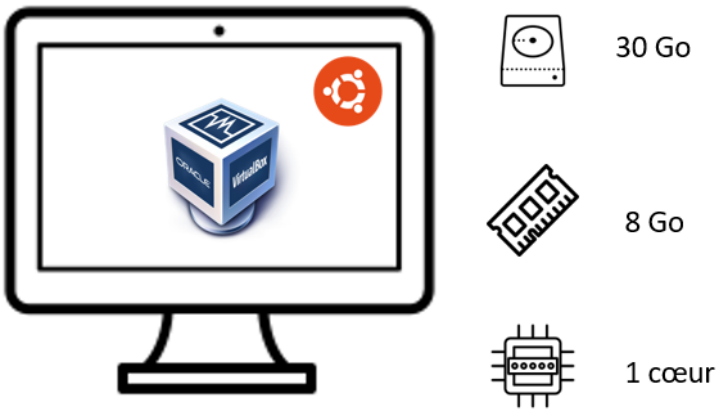


Figure 2 : Environnement technique

L'environnement technique utilisé pour l'installation de la pile ELK stack est la suivante :

Machine virtuelle Virtual Box :

- OS : Ubuntu 20.04
- 30 Go de mémoire de stockage
- 8 Go de mémoire de RAM
- Processeur 1 cœur

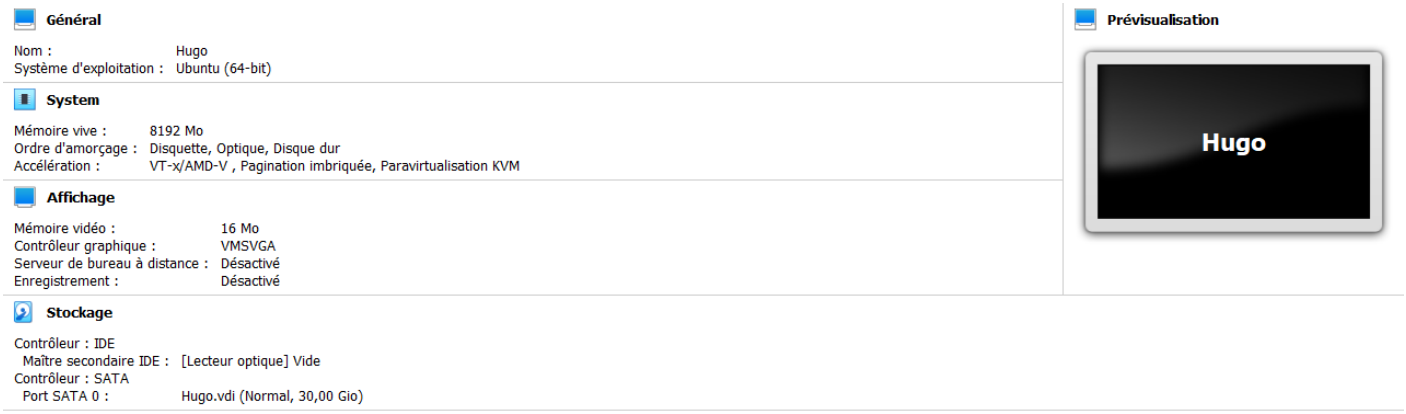


Figure 3 : Caractéristiques technique de la VM

J'ai fait le choix d'allouer à ma VM la moitié de la mémoire RAM disponible sur mon ordinateur personnelle soit 8 Go sur les 16 Go, car la pile ELK est assez gourmande en puissance calculatoire, surtout lorsqu'il s'agit d'exécuter des opérations de filtrage sur les données et ensuite visualiser ces derniers sur le dashboard Kibana. Nous verrons par la suite que la configuration de ma VM est malgré tout limitée en terme de ressources calculatoires.

3. La pile ELK

3.1 Les motivations de la pile ELK

La pile ELK stack est une combinaison de 3 projets open source : ElastricSearch, Logstash et Kibana. Ces briques logicielles ont tous été développés par l'entreprise Elastic pour le stockage et l'analyse des fichiers de logs réseaux. En fait, ELK est conçu pour permettre aux utilisateurs d'analyser et visualiser en temps réel tous types de données réseau et ce quel que soit le format des données collectées.

Une gestion des logs efficace et robuste est devenu vital pour surveiller l'état de fonctionnement l'ensemble des applications du parc informatique d'une entreprise. Pour ce faire, il convient de centraliser les logs générés par toutes les applications en un point de convergence et ainsi faciliter l'analyse des logs.

Originellement, la collecte et la vérification des logs étaient réalisées en se connectant à toutes les machines une à une. Cette méthode étant chronophage, répétitive et fastidieuse, elle n'est plus applicable de nos jours car les systèmes contiennent dorénavant énormément de machines interconnectées. En effet, dans les systèmes cloud il est plus que jamais nécessaire d'automatiser la collecte, le stockage et la vérification des logs.

De nos jours, les architectures orienté micro-services déployés dans le cloud sont très répandus. Or ce genre d'architecture composée de multiples applications et évoluant de manière indépendante génèrent des tonnes de logs. Vérifier manuellement les logs au cas par cas pour chaque application n'est pas envisageable. En effet, cela ne peut pas être fait dans des environnements constitués de centaines de conteneurs générant des To de données de logs par jour. C'est pourquoi, dans les infrastructures cloud il est important pour les administrateurs système de connaitre précisément les performances et l'état de santé des serveurs (ou machines virtuelles). En effet, ces performances peuvent varier en fonction de plusieurs facteurs externes comme par exemple la charge d'utilisation totale du système. La principale motivation de l'introduction de la pile ELK concerne donc la gestion automatisée des flux réseaux permettant aux administrateurs systèmes de prendre les meilleures décisions pour assurer la fiabilité, une bonne disponibilité du système ainsi que pour mieux réagir face à la défaillance des nœuds d'un cluster de VM.

La pile ELK permet de répondre aux besoins des administrateurs en fournissant les fonctionnalités essentielles pour l'automatisation de collecte et la visualisation des log. Ainsi ELK propose :

- Collecte : La collecte des logs provenant de nombreuses sources
- Le traitement : Les logs ne peuvent pas être directement analysées dès lors qu'ils viennent d'être collectées car il est impossible d'effectuer des opérations de filtrage sur ces dernières.
- Stockage : Les données sont collectées et centralisé à un endroit précis
- Indexation : L'indexation permet de faciliter la recherche d'un log. De plus cela est nécessaire pour le requêtage des données
- Visualisation : La création de dashboard facilite la compréhension et l'analyse des logs permettant ainsi de prendre de bonnes décisions assez rapidement.

Dans la suite de ce rapport nous passerons en vue le rôle de chacune des briques logicielles de la pile ELK.

3.2 ElasticSearch

ElasticSearch est une base de données NoSQL orientée document. Celle-ci est basée sur le moteur de recherche Lucene qui permet d'indexer et de chercher du texte très rapidement. ElasticSearch dispose d'une architecture distribuée et stocke les données sous le format JSON et ce sans avoir besoin de définir préalablement un schéma de tables. L'avantage d'ElasticSearch repose sur sa capacité à rechercher en temps réel et rapidement des gros volumes de données. ElasticSearch offre également la possibilité à l'utilisateur d'appliquer des filtres sur les données. De plus, ce dernier est facile à utiliser et évolutif. Au sein de l'architecture ELK, ElasticSearch est en charge du stockage et de l'indexation des données.

Pour la couche logicielle de ElasticSearch ce qui nous intéresse dans le cadre du projet concerne deux éléments :

- **Index :** Un index est une partition logique contenant des documents, semblable à une table dans une base de données relationnelle.
- **Document :** Les documents sont l'unité de stockage de base dans Elasticsearch. Un document JSON contenant des clés et des champs de différents types stocke les informations pertinentes.

3.3 Kibana

Kibana est un outil de visualisation de données qui complète la pile ELK. Il est utilisé pour explorer et visualiser les données ingérées par ElasticSearch. Il permet de rédiger une synthèse visuelle des informations extraites sur les données. Le tableau de bord de Kibana offre la possibilité à l'utilisateur de personnaliser entièrement son dashboard en proposant diverses visualisation interactives (pie chart, bar chart, line chart, carte géographique,…). Par ailleurs Kibana est également utilisé pour réaliser des requêtes complexes et afficher le résultat de cette dernière sous une forme plus compréhensible et intelligible. Kibana dispose également de fonctionnalités de recherche avancées pour affiner, filtrer, agréger les données stockées dans la base ElasticSearch.

3.4 Logstash

Logstash est un outil open source d'intégration de données. Logstash établit un pipeline de collectes de données et alimente la base de données Elasticsearch. Le rôle de logstash est d'unifier plusieurs types de données provenant de différentes sources et de normaliser ces derniers pour les rendre disponibles immédiatement pour une utilisation ultérieure. Pour ce faire, logstash effectue une opération de « parsing » pour extraire les informations contenues dans les données brutes puis une opération de nettoyage pour faciliter l'analyse et la visualisation des connaissances dans kibana.

Dans le cadre de ce projet 3 composantes de logstash nous intéressent à savoir :

- Entrée : Transmission des logs pour les traiter dans un format compréhensible par la machine
- Filtres : Ensemble de conditions pour effectuer une action ou un événement particulier
- Sortie : Restitution de la données nettoyés, filtrées et agrégées.

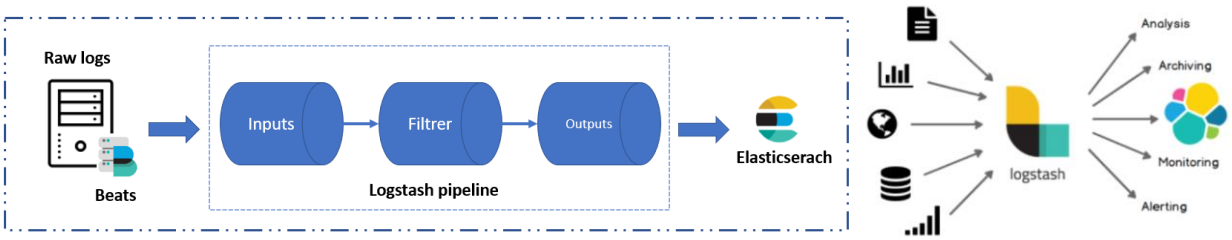


Figure 4 : Logstash pipeline

Le principal avantage de Logstash est que ce dernier propose des filtres prédéfinis et la prise en charge de plus de 200 plugins facilitant l'intégration des données indépendamment de leur source ou leur type.

3.5 Packetbeat

Packetbeat est un agent de la suite Beats permettant l'envoi des logs. Plus concrètement, Packetbeat est capable de collecter des fichiers de capture réseau. L'objectif de Packetbeat est de parser les données contenues dans un fichier de capture réseau afin d'extraire les informations importantes sous un format compréhensible pour ElasticSearch. Cet outil de pipeline de données est souvent utilisé en supplément de Logstash lorsqu'il s'agit d'effectuer des opérations de nettoyage et d'agrégation de données plus complexes avant de pouvoir les charger dans la base ElasticSearch. En fait, packbeat permet de soulager la peine de Logstash en effectuant le plus gros du travail de parsing des données.

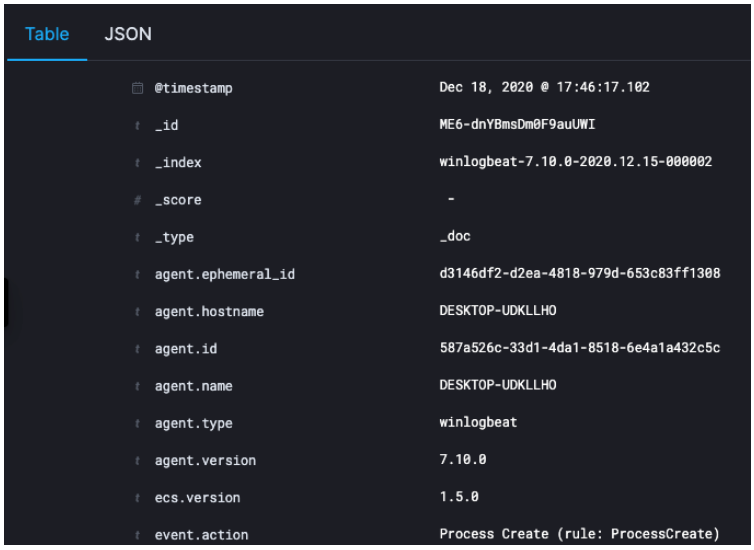


Figure 5 : Parsing des données de fichier de capture réseau par Packetbeat

Source : https://www.lemondeinformatique.fr/actualites/lire-aws-va-forker-elasticsearch-et-kibana-81734.html

3.6 Architecture ELK

L'architecture ELK introduit les fonctionnalités suivantes :

- **Collecte et Agrégation :** La capacité de collecter et d'extraire auprès de multiples sources de données différentes des informations contenus dans les fichiers de capture réseau.

- **Traitement :** La possibilité de transformer les logs réseau dans un format de données compréhensible pour une analyse facile.

- **Stockage :** La possibilité de centraliser à un endroit les données collectées qui seront utilisées ultérieurement pour l'analyse et la surveillance de l'utilisation des applications à des fins de sécurités informatique.

- **Analyse :** La capacité de faire parler les données en les interrogeant et en créant des visualisations et des tableaux de bord interactifs.

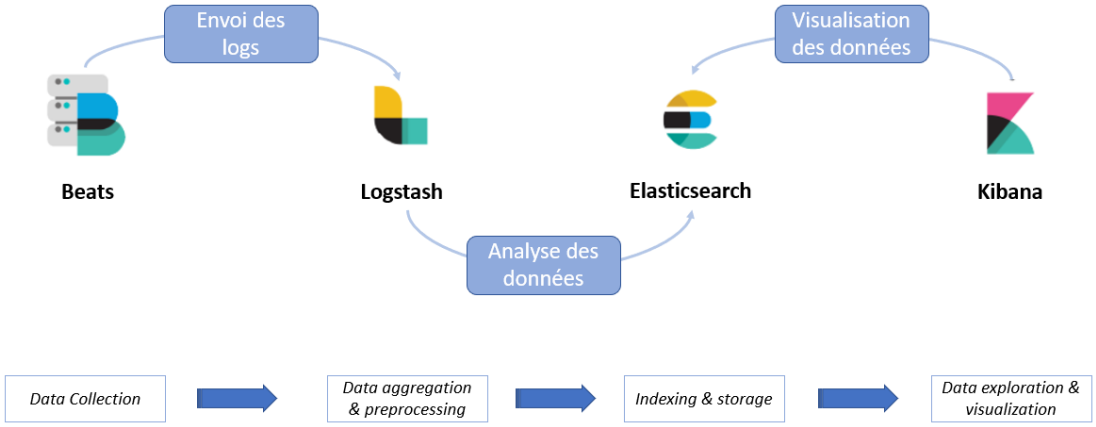


Figure 6 : Architecture ELK

4. Mise en place de la stack ELK

4.1 Installations

Cette section du rapport est consacrée aux étapes de l'installation de la pile ELK. L'idée n'est pas ici de ne pas détailler toutes les commandes bash entrées dans l'invite de commande pour l'installation de la pile ELK. En revanche il s'agira de dresser les étapes les plus importantes de l'installation et notamment la configuration des ports qui seront utilisés par chacune des composantes de la pile ELK.

A noter que l'installation est adaptée au système Linux sous Ubuntu.

La première étape pour la mise en place d'une pile ELK concerne l'installation des dépendances. La pile ELK utilise Java 8 pour fonctionner. Ensuite, il est également nécessaire d'installer de Nginx qui fonctionne comme un serveur et nous permet d'obtenir un accès contrôlé par mot de passe au tableau de bord Kibana.

Installation des dépendances

1. **Installation de java 8**
   sudo apt-get install openjdk-8-jdk
2. **Installation de Nginx**
   sudo apt-get install nginx
3. **Ajout d'un repository Elasticsearch**
   Les répository d'Elasticsearch permettent d'accéder à tous les logiciels open-source de la pile ELK. Pour ce faire, il faut importer la clé GPG.
   wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add –

4. **Installation du package atp-transport-https**
   sudo apt-get install apt-transport-https
5. **Ajout le répository Elastic à la liste répository du système**
   echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee –a /etc/apt/sources.list.d/elastic-7.x.list

La prochaine étape concerne la mise en place d'une pile ELK qui requiert l'installation de 4 composants :

- Installation d'ElasticSearch
- Installation de Logstash
- Installation de Kibana
- Installation de Packetbeats

L'installation de ces composantes suivent des étapes très similaires et la configuration des paramètres qui seront utilisés par ces derniers s'effectue en éditant le fichier de configuration ***.yml*** de chaque brique logicielle.

Installation de la brique ELK (elasticsearch, kibana, logstash)

Installation de la brique ELK en question
sudo apt-get install <elk_block> (ex: sudo apt-get install elasticsearch)

Modification du fichier de configuration .yml
sudo nano /etc/<elk_block>/<ELK_block>.yml (ex : sudo nano /etc/elasticsearch/elasticsearch.yml)

On répète ces lignes de commandes pour l'installation des briques logicielles Elasticsearch, Kibana et Logstash)

Concernant la configuration d'Elasticsearch j'ai modifié le fichier ***elasticsearch.yml*** pour définir le port utilisé par Elasticsearch ainsi que le format du cluster Ealsticsearch (single-node ou mutli-node) et enfin les ressources mémoires allouées par ElasticSearch à la machine virutelle Java :

elasticsearch.yml

```
network.host: localhost
http.port: 9200
discovery.type: single-node
-   Xms4g
-   Xmx4g
```

J'ai choisi d'allouer 4Go à la JVM c'est-à-dire la moitié de RAM disponible de ma VM. De plus la pile ELK sera exécutée en local et non sur un serveur. Par ailleurs j'ai choisi d'utiliser elasticsearch en mode single node car l'idée du projet n'est pas ici de réaliser un stockage de données résilients des données. Toutefois au lancement du service Elasticsearsh, j'ai rencontré une erreur de timeout (voir annexe) interrompant le lancement du service au bout d'un certain temps. Cela est dû aux caractéristiques techniques limitées de ma VM. Pour surmonter ce problème j'ai augmenté le temps de timeout au lancement la brique elasticsearch en éditant le fichier ***Elasticsearch.service*** :

Elasticsearch.service

Modification du fichier elasticsearch.service
sudo nano /usr/lib/systemd/system/elasticsearch.service

TimeoutStartSec=300 (timeout à 300 sec)

On applique les changements
sudo /bin/systemctl enable elasticsearch.service

De manière analogue à l'installation d'ElasticSearch , le fichier ***kibana.yml*** de l'outil Kibana est édité et la configuration se présente comme suit :

kibana.yml

```
server.port: 5601
server.host: "localhost"
elasticsearch.hosts: ["http://localhost:9200"]
```

Après avoir installé et configuré Kibana, j'ai rencontré une erreur lors de son lancement car le pare-feu UFW de ma VM Ubuntu bloquait le trafic sur ce port. Pour résoudre ce problème, j'ai autorisé le trafic sur le port utilisé par Kibana (5601) pour pouvoir accéder au dashboard Kibana via l'url : http://localhost:5601 Ainsi j'ai exécuté la commande ci-dessous :

```
sudo ufw allow 5601/tcp
```

La prochaine étape consiste à installer Packetbeats qui nous permettra de collecter et parser les données contenues dans les fichiers de captures réseau ***PCAP***. Après avoir télécharger et décompressé l'archive d'installation de Packetbeats j'ai apporté des modifications au fichier ***packetbeat.yml.*** J'ai commenté la ligne ***output.elasticsearch*** et décommenté la ligne ***output.logstash*** en y ajoutant le port de logstash***.*** Cela permet de spécifier à l'outil que les données collectées doivent être envoyé à Logstash pour que ce dernier puisse effectuer les opérations de traitement sur les données.

Installation de packetbeat

Récupération de l'archive d'installation de packetbeat

```
curl -L -O https://artifacts.elastic.co/downloads/beats/packetbeat/packetbeat-8.0.0-linux-x86_64.tar.gz*
```

Décompression de l'archive d'installation

```
tar xzvf packetbeat-8.0.0-linux-x86_64.tar.gz
```

packetbeat.yml

```
setep.kibana:
host:"localhost:6501"
output.logstash:
host: ["localhost:5044"]
```

Il convient maintenant d'installer Logstash. Pour ce faire, j'ai créé un fichier ***test.conf*** pour y définir, le port d'entrée qui est celui utilisé par Packetbeats et le port de sortie utilisé par Elasticsearch. Ce fichier est nécessaire pour définir la configuration du pipeline de données nécessaire pour la l'extraction, la transformation et le chargement des données dans Kibana. Avec ce fichier, Logstash comprend qu'il doit récupérer les données collectées par PacketBeats et doit ensuite les envoyer à ElasticSearch pour le stockage des données prétraitées. Le fichier ***test.conf*** se présente comme ci-dessous :

test.conf

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

output {
  elasticsearch  {
```

```
        hosts => ["http://localhost:9200"]
    }
}
```

La lecture d'un fichier de données ***PCAP*** par le pipeline de données peut être lancé via la commande suivante :

```
./packetbeat -e -c packetbeat.yml -I <fichier.pcap>
```

La séquence des commandes a exécuter pour le démarrage de chacune des composantes de la pile ELK est décrite en annexe. C'est avec cette séquence de commandes que nous lançons le processus d'intégration des données dans Elasticsearch.

4.2 Description du fichier PCAP

PCAP est une extension de fichier associé principalement au logiciel Wireshark qui permet d'analyser les paquets de données. Ainsi les fichiers ***PCAP*** sont des fichiers de données qui contiennent les paquets de données qui transitent au sein d'un réseau. Ces fichiers sont utilisés pour surveiller le trafic d'un réseau et détecter des anomalies au sein d'un réseau.

J'ai volontairement choisi un fichier ***PCAP*** très léger car après plusieurs tentatives avec d'autres fichiers ***PCAP*** plus volumineux le chargement des données dans l'interface Kibana était très long et entrainé un « chrash » de ma VM.

Pour toutes ces raisons, le fichier choisi est un fichier ***PCAP*** provenant d'un réseau de systèmes de contrôle industriels. Ce fichier est disponible ici.

4.3 Workflow du pipeline de données

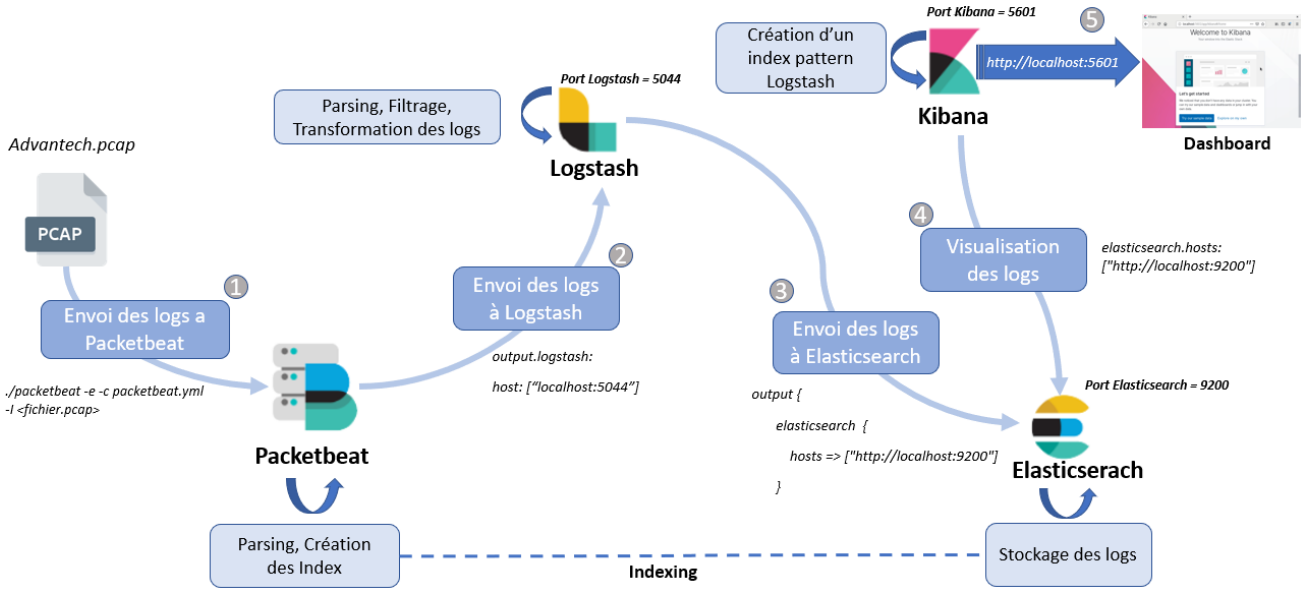


Figure 7 : Pipeline de données

4.4 Création d'un index sur Kibana

Kibana nécessite un modèle d'index pour accéder aux données stockées dans Elasticsearch. Ainsi avant de pouvoir visualiser les données il convient de créer ce que l'on appelle un « index pattern » directement dans Kibana. Cette

index pattern permet de spécifier à Kibana quel modèle de données suivent les données que l'on souhaite visualiser. Dans notre cas, les données ont été traitées par logstash. Par conséquent le modèle de données est celui défini par logstash.

Pour la création d'un « index pattern » dans Kibana, il faut se rendre l'onglet « Stack Management » puis « Index Pattern». Depuis la page Index Pattern, il est possible de créer un index logstash. La création d'un index logstash depuis l'interface de Kibana est décrite comme suit :

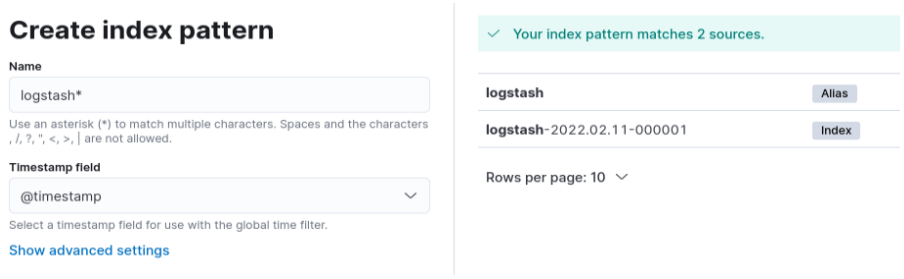


Figure 8 : Création d'un index logstash dans kibana

5. Dashboard Kibana

5.1 Discover

Avant de débuter la réalisation du tableau de bords, j'ai souhaité en premier lieu me familiariser avec les données en les explorant via l'outil ***Discover*** de Kibana. En effet cet outil facilite l'exploration des données. Discover permet d'avoir un aperçu rapide sur les données indexées par logstash. Par ailleurs, il est possible d'obtenir des informations sur la structure des champs de notre jeu de données. Il est également possible d'appliquer des filtres afin de sélectionner certaines données de notre jeu de données et de les visualiser. Par exemple on peut choisir de visualiser les données dont le champs « timestamp» respecte une plage horaire spécifique. En résumé, ***Discover*** permet de requêter sur les données, filtrer le résultat de la requête et afficher les données d'un document en particulier.

L'interface de l'outil ***Discover*** se présente comme ci-dessous :

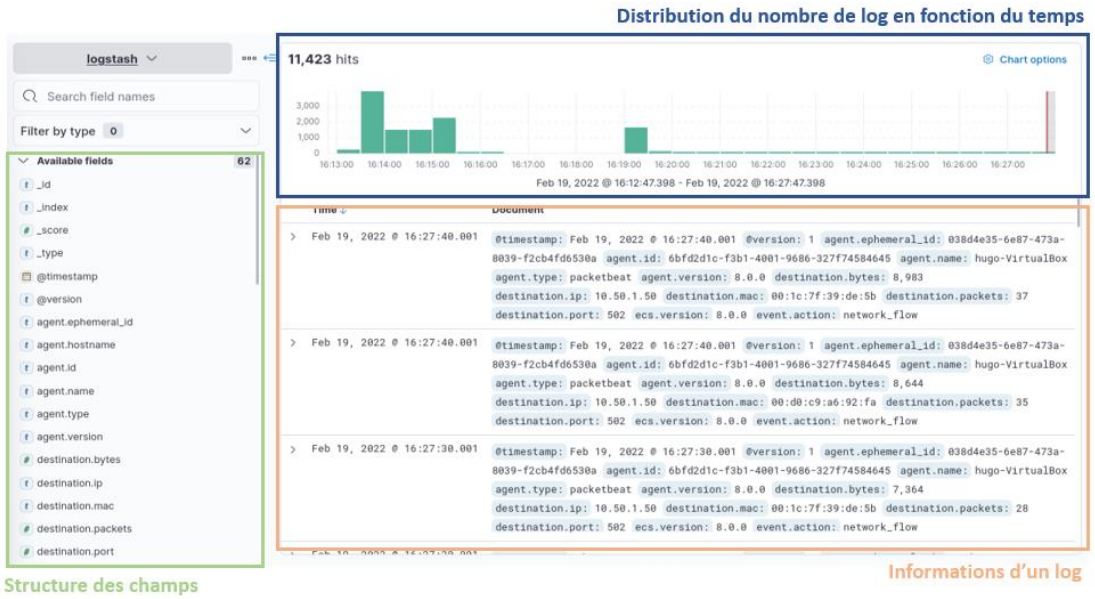


Figure 9 : Fonctionnalité Discover de Kibana

Sur la figure ci-dessous, nous avons une série de champs indexées par Logstash. Par exemple, on y trouve le champs ***Timestamp*** qui correspond à la date à laquelle l'évènement a été capture sur le réseau. On retrouve également plusieurs champs propres aux paquets de données tel que : ***destination.ip, destination.mac*** qui correspondent respectivement à l'adresse ***ip*** à laquelle ce paquet de données est destiné et l'adresse ***mac*** à laquelle ce fichier est destiné.

Pour construire le dashboard Kibana, je me suis largement appuyé sur la fonctionnalité ***Discover*** de Kibana afin d'y sélectionner les données intéressantes à visualiser. Une fois les données sélectionnées et filtrées, l'outil ***Discover*** nous propose de visualiser directement les données sélectionnées comme présenté ci-dessous.

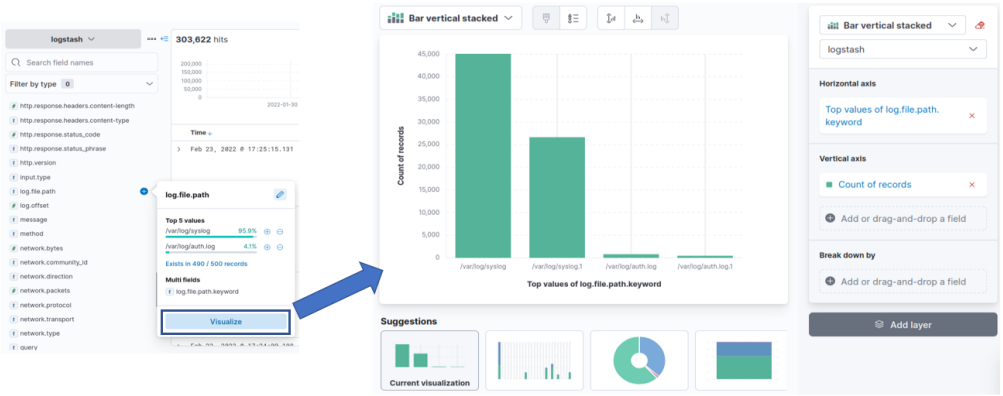


Figure 10 : Passage de fonctionnalité Discover à la fonctionnalité Dashboard

5.2 Le dashboard Kibana

L'outil dashboard de Kibana offre la possibilité de combiner plusieurs visualisations des données de logs construites à partir des index crée par Logstash. Les visualisations sont interactives dans la mesure où il est possible de survoler le graphique pour afficher la valeur exacte d'une donnée en particulier. L'avantage de l'utilisation de dashboard Kibana est qu'il est possible de mettre à jour automatiquement l'ensemble des graphiques en quasi-temps réel. Par ailleurs, cet outil offre plusieurs types de visualisations possibles tel que le diagramme en bâtons, diagrammes à barres, camemberts, courbes chronologiques, cartographie, compteurs,...

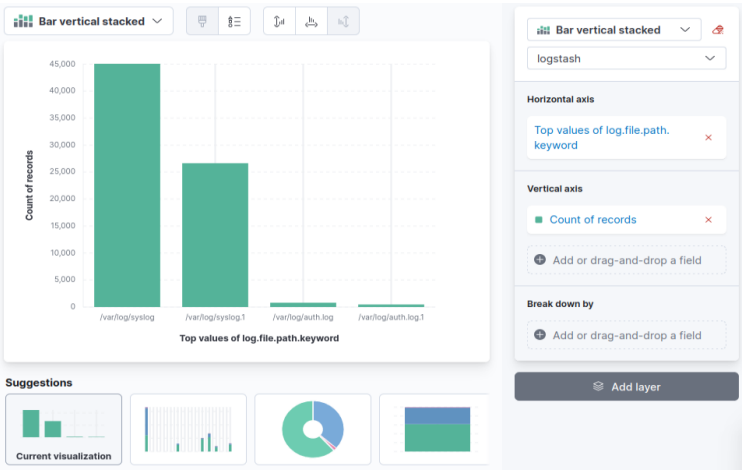
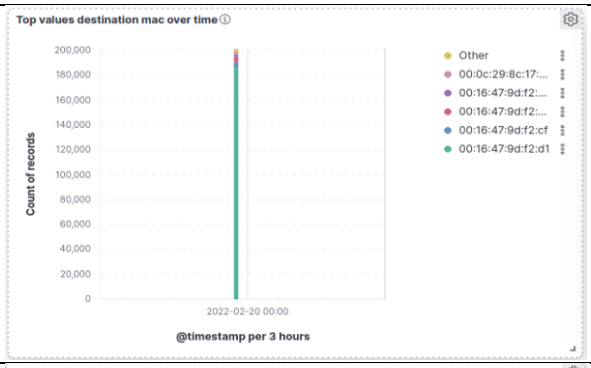
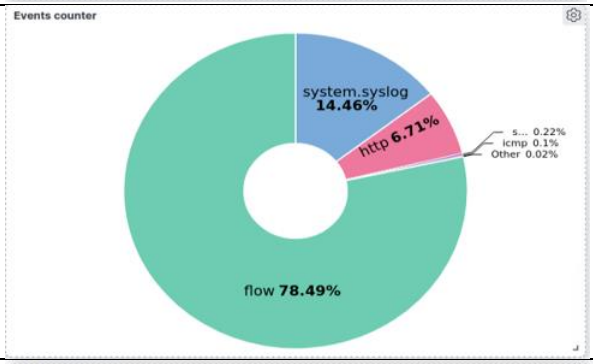
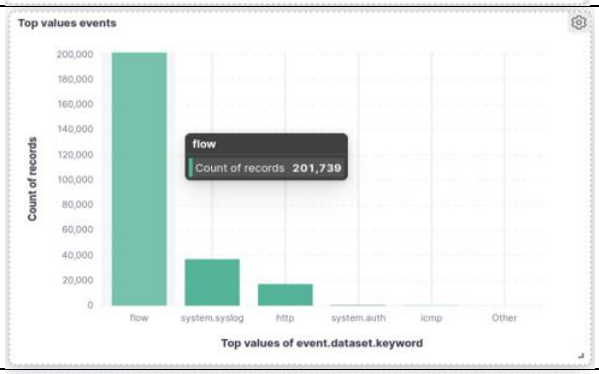
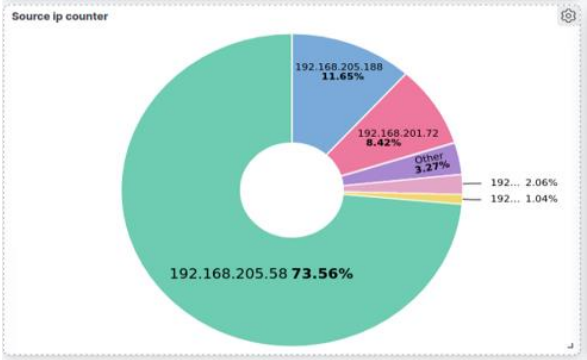
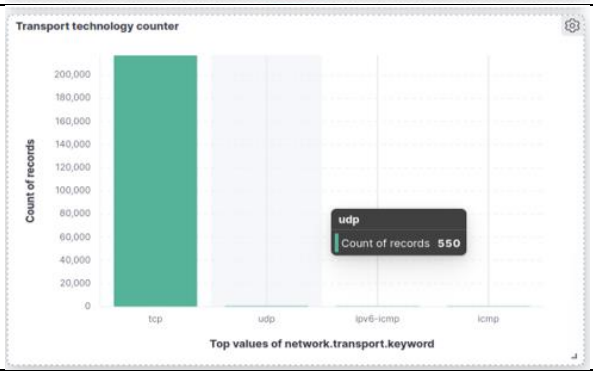
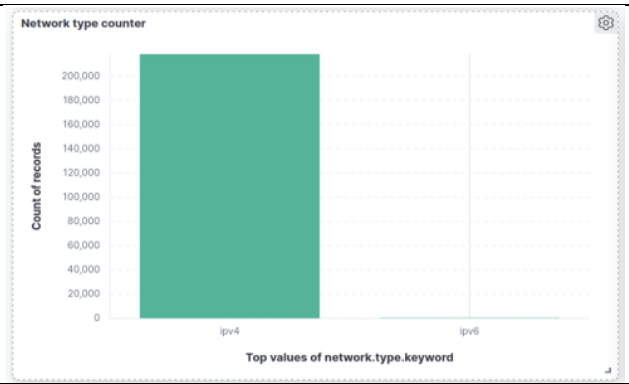


Figure 11 : Interface de la création d'un visuel pour le dashboard

En construisant le dashbord de ce projet j'ai choisi d'afficher les métriques suivantes :

Destination mac Visualisation de l'évolution du nombre d'adresse mac vers lesquels un paquet de données est envoyé en fonction du temps	
Events type Visualisation du nombre d'évènement du log en fonction de son type (system, http, flow,…)	
Top 5 Events type Visualisation des 5 évènement les plus présents dans notre jeu de données.	
IP source Visualisation du nombre des IP qui sont à l'origine de l'envoi de paquets de données.	
Transport technologie Visualisation du nombre de paquets qui ont été transmis avec un protocole de transmission en particulier sur le réseau (TCP, UDP,ICMP,…)	

Network type Visualisation du nombre de machines connectées à internet depuis le réseau avec une adresse IPv4 ou IPv6.	
Log file path Evolution du nombre de fichiers de log en fonction du temps.	

D'autres visualisations sont disponibles dans le fichier « ndjson ». En important ce dernier dans Kibana il est possible de visualiser le dashboard complet.

6. Conclusion

A travers ce projet dédié à la sécurité informatique adapté au big data, j'ai pu mettre en place une pile ELK pour l'analyse de capture de fichiers réseaux. Ainsi j'ai pu m'apercevoir des avantages offerts par la pile ELK lorsqu'il convient d'analyser une grande quantité de logs provenant de multiples sources. J'ai surtout compris les enjeux de la pile ELK quand il est question pour une entreprise de surveiller en temps réel l'utilisation de ses systèmes informatiques déployés dans le cloud. Par ailleurs j'ai également pu m'apercevoir des limites de la pile ELK qui concerne essentiellement l'utilisation très gourmande en ressource mémoire des briques logicielles Elasticsearch et Logstash. En effet, les 8 Go RAM que j'ai attribué à ma VM n'ont pas été suffisant car à plusieurs reprises, la visualisation des données depuis Kibana à fait crasher ma machine virtuelle. Au même titre que la visualisation des données depuis Kibana ma VM accusé une lenteur phénoménale pour l'intégration des données dans la base Elasticsearch. Malgré ces problèmes rencontrés j'ai pu parvenir à la construction d'un dashboard Kibana. Une des améliorations possibles pour la pile ELK que j'ai mise en place concerne la configuration des paramètres du cluster Elasticsearch pour rendre cette dernière résiliente aux pannes notamment en modifiant la configuration du cluster Eslacticsearch en mode multi-node.

7. Annexe

Démarrage de la pile ELK

Elastic Search
Start Elastic
sudo systemctl start elasticsearch.service
Test Elastic
curl -X GET "localhost:9200"
http://localhost:9200

Kibana
Start Kibana
sudo systemctl start kibana
http://localhost:5601

Logstash
Start Logstash
sudo systemctl start logstash
Check status
sudo systemctl status logstash

Packetbeat
Lancer packet beat (Lecture des fichiers logs)
sudo ./packetbeat -e -c packetbeat.yml -I /home/hugo/Projet_ELK -d

Problème timeout au lancement d'Elasticsearch

```
hugo@hugo-VirtualBox:~$ sudo systemctl start elasticsearch.service
[sudo] Mot de passe de hugo :
Job for elasticsearch.service failed because a timeout was exceeded.
See "systemctl status elasticsearch.service" and "journalctl -xe" for details.
hugo@hugo-VirtualBox:~$
```

8. Bibliographie

AWS [aws.amazon.com], ***La suite ELK***,
[en ligne], [Consulté le 16/02/2022].
Disponible à l'adresse : https://aws.amazon.com/fr/opensearch-service/the-elk-stack/

Thomas Roccia (Security Researcher) [openclassrooms.com], ***Optimisez la sécurité informatique grâce au monitoring***,
[en ligne], [Consulté le 16/02/2022].
Disponible à l'adresse :
https://openclassrooms.com/fr/courses/1750566-optimisez-la-securite-informatique-grace-au-monitoring

Rédacteur de Logs.io [logz.io], ***THE COMPLETE GUIDE TO THE ELK STACK***,
[en ligne], [Consulté le 16/02/2022].
Disponible à l'adresse : https://logz.io/learn/complete-guide-elk-stack/#intro

Rédacteur oxxeo.fr [oxxeo.fr], ***Découverte de la stack Elastic***
[en ligne], [Consulté le 16/02/2022].
Disponible à l'adresse : https://www.oxxeo.fr/post/decouverte-elk