RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.bj Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Sécurité Informatique

Présenté par :
Ariel M. Wilson **SOSSOU LOKO**

Déploiement d'une solution de visualisation du système d'information avec ELK: cas de Soldigit

Sous la supervision :
Ing Mawuénan Hervé Guy **AKAKPO**
Chef Département des Systèmes d'Information à la Caisse Autonome de Gestion de la Dette du Bénin

Membres du jury :

Mr Arnaud AHOUANDJINOU	Docteur	IFRI-UAC	Président
Mme Miranda GNONLONFOUN	Ingénieure	IFRI-UAC	Examinateur
Mr Hervé AKAKPO	Ingénieur	IFRI-UAC	Rapporteur

Année Académique : **2022-2023**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Table des figures vi

Liste des tableaux vii

Liste des acronymes viii

Glossaire 1

Introduction 2

1 Revue de littérature 4

2 Présentation du stack ELK 9

3 Test et Implémentation 16

Conclusion 32

Bibliographie 33

Webographie 34

Table des matières 35

Dédicace

Je dédie ce travail,
A mes parents,
A mes proches et à ceux qui me sont chers,
Pour vos sacrifices et votre soutien tout au long de mon cursus.

Remerciements

Tout d'abord, nous rendons grâce à Dieu de nous avoir donné le courage, l'enthousiasme, la volonté ainsi que la force nécessaire pour la réalisation de ce mémoire.

Nous adressons nos remerciements les plus sincères à nos parents qui tout au long de ce travail, ont apporté leur soutien et leurs encouragements. Nos sincères remerciements s'adressent à M. Eugène EZIN le Directeur de l'Institut, à toute l'administration et au corps professoral pour la qualité de la formation que nous avons reçu durant ces trois années.

Nous voudrions aussi remercier, le maitre de mémoire M. Hervé AKAKPO, pour sa patience, sa disponibilité et l'encadrement apporté tout au long de ce projet de mémoire.

Nous remercions M. Jospy GOUDALO, pour ses idées, sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont aidé dans la réalisation de ce projet de mémoire.

Nos remerciements s'adressent à nos proches pour leur soutien moral et aide durant la rédaction du présent mémoire. Nos vifs remerciements vont également aux membres du jury qui ont accepté d'examiner ce travail et de l'enrichir par leurs observations et remarques.

Résumé

Aujourd'hui, malgré des efforts continus pour renforcer la sécurité des systèmes d'information, les entreprises peinent encore à identifier avec précision l'origine des attaques et la nature des menaces. Pour pallier cette situation, la mise en place d'une solution de supervision informatique devient une action cruciale et indispensable dans la gestion et la protection des systèmes d'information. A l'issue de la comparaison de plusieurs outils, le stack ELK se démarque comme meilleure solution pour la surveillance en temps réel des performances et la détection des anomalies dans le système d'information. L'objectif de ce projet est d'améliorer la visibilité et la réactivité face aux menaces potentielles sur les services de Soldigit. Des simulations et des tests ont été réalisés pour apprécier la solution déployée. Ce travail nous a permis de mettre en place un environnement de supervision fiable et adapté aux besoins de l'organisation.

Mots clés : Sécurité, Attaque, Supervision, Stack ELK.

Abstract

Today, despite ongoing efforts to strengthen information systems security, companies are still struggling to pinpoint the origin of attacks and the nature of threats. To remedy this situation, the implementation of an IT monitoring solution is becoming a crucial and indispensable action in the management and protection of information systems. After comparing several tools, the ELK stack emerged as the best solution for real-time performance monitoring and anomaly detection in the information system. The aim of this project is to improve visibility and responsiveness to potential threats to Soldigit's services. Simulations and tests were carried out to assess the solution deployed. This work enabled us to set up a reliable supervision environment adapted to the organization's needs.

Key words: Security, Attack, Supervision, Stack ELK.

Table des figures

1.1 Architecture de supervision informatique.[3] 5

2.1 Fonctionnement du Stack ELK.[4] 10
2.2 Configuration de la **phase Input** 11
2.3 Syntaxe d'un pattern grok 12
2.4 Configuration de la **phase Output** 12
2.5 Présentation de l'interface web de Kibana. 13

3.1 Architecture de Soldigit 16
3.2 Architecture de Soldigit après déploiement de la solution 17
3.3 Mécanisme de fonctionnement de la solution proposée 18
3.4 Interface Web Elasticsearch après installation 19
3.5 Interface web kibana après installation 19
3.6 Extrait paramètre de configuration elasticsearch 20
3.7 Répertoire de Configuration kibana 21
3.8 Statut actif sur le service filebeat 21
3.9 Statut actif sur le service metricbeat 22
3.10 Statut actif sur le service packetbeat 22
3.11 Fichier de configuration du service filebeat 22
3.12 Fichier de configuration du service metricbeat 23
3.13 Fichier de configuration du service packetbeat 23
3.14 Insertion des données de connexion 24
3.15 Page d'authentification du stack ELK 25
3.16 Collecte d'information log avec filebeat 25
3.17 Collecte d'informations avec l'agent metricbeat. 26
3.18 Représentation des informations de collecte sous forme de graphe. 27
3.19 Création de la règle avec metricbeat. 28
3.20 Réception du message d'alerte avec metricbeat. 28
3.21 Collecte d'informations avec packetbeat 28
3.22 Suite d'informations de collecte packetbeat 29
3.23 Création de la règle avec packetbeat. 29
3.24 Réception du message d'alerte avec packetbeat. 30
3.25 Collecte d'information de connexion avec winlogbeat. 30
3.26 Suite d'information de connexion. 30

Liste des tableaux

2.1 Grille comparative des solutions de supervision. 10

Liste des acronymes

API :

Application Programming Interface

ELK :

Elasticsearch Logstash Kibana

FTP :

File Transfer Protocol

HTTP/HTTPS :

Hyper Text Transfer Protocol/SecureProtocol

IBM :

International Business Machine

NAS :

Network Attached Storage

OS :

Operating System

SNMP :

Simple Network Management Protocol

Glossaire

ELK : Outil d'analyse de logs composé de 3 logiciels : Elasticsearch, Logstash et Kibana. viii

FTP : Protocole de transfert de fichiers par Internet. Il permet l'échange de commandes et de données entre un ordinateur client FTP, et un serveur, l'hôte FTP. viii

HTTP/HTTPS : Est une extension sécurisée du protocole HTTP, le « S » pour « Secured » (sécurisé) signifie que les données échangées entre le navigateur de l'internaute et le site web sont chiffrées . viii

NAS : Appareil de stockage autonome qui peut se connecter à votre réseau privé ou professionnel via Internet. viii

SNMP : Protocole de communication qui permet aux administrateurs réseau de gérer les équipements du réseau, de superviser et de diagnostiquer des problèmes réseaux et matériels à distance. viii

Introduction Générale

Le système informatique est un pilier sensible au développement de toutes les entreprises modernes. Il est essentiel pour ses entreprises de s'approprier leurs architectures réseau afin de mieux réagir face aux menaces. C'est dans une telle perspective que le déploiement des outils de supervision devient une priorité d'envergure pour ces derniers.

1- Problématique

Les temps d'arrêts couteux, la performance médiocre des équipements et applications ainsi que les infrastructures mal supervisées sont des facteurs pouvant mettre en péril la viabilité d'un système d'information. Pour assurer un bon fonctionnement de ses services, une organisation se doit d'optimiser ses ressources, de renforcer sa sécurité et de prendre des décisions éclairées. Avec une vue d'ensemble sur l'actif informationnel, l'administrateur du système d'information garantit ces points abordés.

2- Contexte

Le progrès technologique pousse les entreprises à adopter de nouvelles solutions pour améliorer leurs ressources informatiques. Malgré tous nos efforts pour renforcer nos infrastructures, il arrive fréquemment que des problèmes de disponibilité des services apparaissent pour les utilisateurs. Chez Soldigit, spécialisée dans la fourniture de services web, il est essentiel d'avoir une vue permanente sur nos services pour assurer un maintien continu et efficace des solutions proposées. Cependant, nous avons constaté que les clients se plaignent parfois des services quelque temps après leur livraison. Face à ces plaintes, il devient impératif d'implémenter une solution de supervision dans le but d'optimiser et de renforcer la performance de nos services.

3- Objectif

3.1 Objectif général

L'objectif principal de notre travail est de :

- Garantir une visibilité et une corrélation optimale des données de sécurité en entreprise.
- Faire un déploiement technique de la solution en entreprise.

3.2 Objectifs spécifiques

Il s'agit spécifiquement de :

- Faire une étude des différents outils de supervision existants afin de faire le choix de la solution à implémenter en prenant en compte les besoins ;
- Configurer le stack ELK en prenant en compte les bonnes pratiques ;
- Tester et valider la solution de supervision.

4- Organisation du document

Nous avons structuré notre travail en trois grands chapitres. Le premier chapitre aborde les généralités sur la supervision, c'est-à-dire une étude des différentes solutions et protocoles de supervision disponibles. Le deuxième chapitre porte sur la description du fonctionnement de notre solution et dans le troisième chapitre l'implémentation et la présentation des résultats obtenus suite à des tests de validation.

Chapitre 1

Revue de littérature

Introduction

Dans ce premier chapitre, nous allons présenter les généralités sur la supervision, ensuite nous évoquerons quelques outils de supervision au sein des entreprises pour le déploiement d'une infrastructure de supervision.

1.1 Généralités sur la supervision

Il sera question de définir la supervision, montrer ses objectifs, son importance et son fonctionnement dans le système informatique.

1.1.1 Définition et objectif

La supervision informatique, également appelée monitoring du système informatique, est un processus **qui assure la surveillance et la gestion en temps réel des systèmes, des processus et les activités critiques d'une organisation afin d'être au courant de l'état du système informatique**. En d'autres termes, elle se définit comme un processus qui consiste à recueillir des données sur le fonctionnement du matériel et des logiciels d'un environnement informatique afin de s'assurer que tout fonctionne comme prévu. En un mot, la supervision permet d'avoir un état de santé du système d'information. Son objectif principal est d'identifier à temps les anomalies liées aux infrastructures afin de prendre des décisions éclairées pour optimiser la ressource.

1.1.2 Importance de la supervision

La supervision se positionne comme une solution idéale et indispensable à la proactivité d'une entité. En effet, elle permet à l'administrateur système de réagir plus vite et de manière efficace aux incidents, de garantir un niveau de service maximal et de renforcer dans le même temps la sécurité informatique.

1.1.3 Fonctionnement de la supervision

La solution de supervision se base généralement sur une vue globale du fonctionnement du système d'information, en particulier sur les services du parc informatique. Elle se base sur quatre étapes essentielles qui sont la collecte, le traitement, l'analyse et la présentation de la donnée.
Les données sont **collectées** à partir de différentes sources telles que les équipements réseaux, les serveurs, les applications, les bases de données, etc. Ces données peuvent inclure des logs, des métriques, des évènements, des performances et des états exploitables pour l'optimisation de l'infrastructure. L'étape de **traitement** est le niveau de normalisation où sont appliquées les règles et les filtres préalablement définis pour rendre les informations cohérentes, compatibles et faciles à interpréter. Ces données sont ensuite stockées dans une base afin d'être **analysées** sur des critères bien définis, des techniques comme celle de corrélation, de tendance, de détection d'anomalies. Afin que l'information soit utile lors de sa visualisation ou sa **présentation** sous forme de tableaux de bord, de graphiques, de diagrammes, etc.

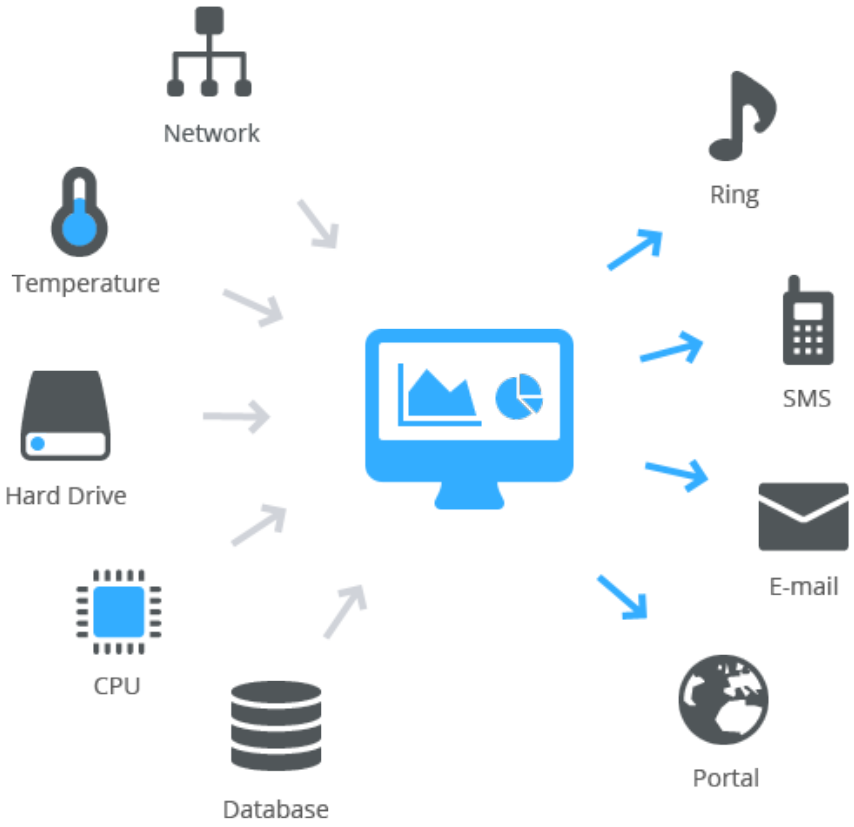


FIGURE 1.1 : Architecture de supervision informatique.[3]

1.1.4 Protocoles de supervision

Pour le bon acheminement des informations au sein de la plateforme de supervision, des protocoles sont mis en jeu. Ces protocoles permettent d'établir des communications entre les éléments à superviser et les systèmes de gestion de la supervision, facilitant ainsi la collecte et l'analyse des données pour garantir la performance et la disponibilité des systèmes informatiques.

Parmi les protocoles couramment utilisés dans la supervision informatique, nous trouvons le **Simple Network Management Protocol (SNMP)**. SNMP est l'un des protocoles les plus utilisés pour la supervision des équipements réseau. Décrit dans la RFC1157 par l'Internet Architecture Board

(IAB), ce protocole de couche application est intégré à la plupart des équipements réseau, permettant la gestion, la supervision et le diagnostic des problèmes réseau et matériel à distance. SNMP se compose de trois éléments essentiels : le Gestionnaire SNMP, l'Agent SNMP, et la base de données d'informations de gestion (MIB).

Un autre protocole important est **Syslog**, qui standardise l'envoi de messages de journalisation et d'alertes générées par les équipements réseau et les applications vers un serveur centralisé. Ce protocole, défini dans le RFC 3164 et utilisant le port 514 de l'UDP, permet de centraliser les logs pour une analyse plus efficace des événements.

L'**Internet Control Message Protocol (ICMP)** est spécifiquement conçu pour les rapports d'erreurs et la surveillance de base. Utilisé pour des tâches telles que les pings et les tracerts, ICMP aide à signaler des conditions d'erreur et à sonder le réseau pour en déterminer les caractéristiques générales.

L'**HTTP/HTTPS (Hypertext Transfer Protocol/Secure)**, développé par Tim Berners-Lee, permet l'accès aux interfaces web des dispositifs, des applications ou des outils de supervision. Ce protocole facilite l'interaction avec les systèmes pour récupérer des données et configurer des paramètres, ajoutant une couche de flexibilité à la supervision informatique.

Les **Flow Protocols** tels que NetFlow, sFlow et IPFIX sont utilisés pour collecter et analyser des données de flux réseau. Par exemple, NetFlow, un protocole réseau propriétaire de Cisco, permet de surveiller le trafic réseau et d'identifier les tendances et les problèmes de performances en envoyant des enregistrements de flux réseau à un outil de collecte centralisé.

Dans le domaine de l'automatisation industrielle, le protocole **Modbus** joue un rôle crucial. Il définit un protocole de transmission de messages pour les communications «client-serveur» entre des dispositifs reliés par un bus ou un réseau, facilitant ainsi la supervision et le contrôle des équipements et des processus.

Enfin, les **Java Management Extensions (JMX)** offrent une API Java pour la supervision et l'administration d'applications. JMX permet de collecter des données de performance, de générer des statistiques en temps réel et de notifier des dysfonctionnements, contribuant ainsi à une gestion efficace des ressources.

En résumé, ces protocoles jouent un rôle vital dans la supervision informatique en assurant la communication, la collecte de données, l'analyse et la gestion des systèmes pour garantir leur performance et leur disponibilité.

1.1.5 Type de supervision[2]

L'infrastructure informatique peut être surveillée à plusieurs niveaux, et le type de supervision joue un rôle crucial dans cette surveillance. Il est donc essentiel de bien comprendre les différents types de supervision disponibles.

La première forme de supervision est la **supervision système**. Celle-ci veille au bon fonctionnement des systèmes d'exploitation. Elle contrôle les messages d'erreur, met à jour les composants des systèmes d'exploitation, et répertorie le nombre de processus actifs. En assurant ces tâches, elle garantit que les systèmes sont opérationnels et performants.

Ensuite, nous avons la **supervision service**. Elle se concentre sur le maintien du bon fonctionnement des services web, des sauvegardes automatiques, des bases de données, ainsi que des serveurs NAS ou FTP. En d'autres termes, elle assure la disponibilité et la fiabilité des services essentiels à l'activité de l'entreprise.

La **supervision réseau** est un autre type crucial de supervision. Elle surveille les différentes composantes du réseau informatique, y compris l'accessibilité des routeurs, la qualité des bandes passantes, et le bon fonctionnement des ports. Ainsi, elle garantit une connectivité réseau stable et efficace.

Enfin, la **supervision applicative** se focalise sur les applications. Elle évalue leur disponibilité, leur performance, et l'intégrité des données qu'elles manipulent. En surveillant ces aspects, elle assure que les applications fonctionnent correctement et répondent aux besoins des utilisateurs.

1.1.6 Les avantages de la supervision[9]

Le déploiement d'une solution de supervision offre de nombreux avantages à l'administrateur informatique, constituant un gain de temps précieux.

Premièrement, la supervision assure une **disponibilité continue et efficace des ressources**. En surveillant en permanence les systèmes et services, elle permet d'identifier rapidement les problèmes potentiels avant qu'ils ne deviennent critiques.

Deuxièmement, elle permet une **résolution prompte et proactive des problèmes**. Grâce à une détection précoce des anomalies, les interventions peuvent être effectuées avant que les utilisateurs ne soient affectés, réduisant ainsi les interruptions de service.

De plus, la supervision contribue à **prolonger la durée de vie des machines**. En prenant en charge les menaces et en effectuant des maintenances régulières, elle prévient l'usure prématurée des équipements.

Ensuite, elle participe à **maintenir une sécurité renforcée et à améliorer la performance du système**. En surveillant les activités suspectes et en optimisant les ressources, la supervision renforce la protection contre les cyberattaques et assure des performances optimales.

En outre, elle offre une **meilleure capacité à faire face aux attaques**. En détectant rapidement les tentatives d'intrusion, elle permet de réagir immédiatement et de limiter les dommages potentiels.

Enfin, la supervision améliore le **contrôle des processus de surveillance**. En centralisant les informations et en automatisant les tâches répétitives, elle facilite la gestion et la surveillance des systèmes informatiques.

1.2 Les différents types de solution de supervision

Il existe différents types de solution de supervision qui varient en fonction de leurs fonctionnalités, de leurs caractéristiques et de leur déploiement. Ces solutions sont classées en deux grandes catégories dont :

- **Solution Propriétaire**
  Une solution propriétaire est un logiciel qui appartient uniquement à la personne morale ou physique qui l'a développé. Contrairement à un outil open source, une solution propriétaire et son code source ne peuvent être étudiés, modifiés ou distribués sans autorisation formelle de son créateur ou éditeur.

- **Solution Open-Source[7]**
  Une solution open-source ou « code source ouvert » fait référence à un logiciel dont le code

source est mis gratuitement à la disposition de tous avec les possibilités d'y accéder, de le modifier et de le redistribuer selon les critères établis par l'Open-Source Initiative (OSI). En effet, ce code source peut alors être exploité par les développeurs et les entreprises souhaitant soient, l'adapter à leurs besoins métiers, soit affiné son intégration avec leur système d'information.

1.3 Etude comparative des solutions propriétaires et open-source de supervision

Comme dit précédemment, nous distinguons deux grandes catégories de solutions de supervision, lesquelles présentent bien entendu des avantages, mais ne sont pas pour autant exemptes de quelques limites.
La prochaine étape de notre travail a donc consisté à sélectionner quelques-unes des solutions de chacune des catégories ci-dessus énumérées, afin de faire leur étude comparative et par la suite choisir une solution qui répond au besoin de l'entreprise. Notre étude est faite suivant les critères ci-après : **coût, sécurité, système d'exploitation hôte, limitations (quantité de logs ou événements) et catégorie (Open-Source ou propriétaire)**.

Les solutions de supervision retenues pour notre étude sont les suivants :

- Splunk Entreprise
  Splunk Entreprise est une solution puissante pour la gestion des données et la sécurité informatique, offrant une combinaison de collecte de données en temps réel, d'analyse avancée, de détection des menaces, de gestion des incidents et de conformité réglementaire.

- QRadar
  QRadar est une solution propriétaire conçu pour contrer les menaces tout en améliorant les opérations de sécurité. Il offre une visibilité centralisée sur les données de sécurité à l'échelle de l'entreprise et fournit des informations exploitables sur les menaces les plus urgentes. Grâce à ses fonctionnalités intégrées de surveillance, de détection, d'enquête et de réponse, QRadar permet une approche holistique de la sécurité, aidant ainsi les organisations à gérer efficacement les menaces et à protéger leurs actifs numériques.

- Nagios
  Nagios est une solution robuste et flexible pour la surveillance des systèmes informatiques, offrant une visibilité essentielle sur la santé et les performances de l'infrastructure, et facilitant la détection proactive des problèmes et la prise de mesures correctives.

Conclusion

Dans cette section, nous avons montré les généralités sur la supervision. Ensuite, nous avons présenté quelques-uns de ses outils dont le choix s'est fait sur la base des besoins et objectifs des entreprises. Dans le chapitre qui suivra, nous mettrons l'accent sur la présentation de notre choix.

Chapitre 2

Présentation du stack ELK

Introduction

Avant de mettre en place notre système de supervision, il importe d'avoir une connaissance approfondie sur ce dernier et de connaître son fonctionnement. Comme annoncé, nous allons présenter le fonctionnement, les caractéristiques ainsi que quelques actions et commandes du stack ELK (Elasticsearch, Logstash et Kibana).

2.1 Choix de la solution de supervision

2.1.1 Justification et critère du choix pour l'utilisation d'une solution de supervision à Soldigit

Pour une entreprise comme Soldigit, spécialisée dans les services web, disposer d'une solution de supervision est essentiel pour assurer la disponibilité, la performance et la sécurité de ses services. Une telle solution permet de surveiller en temps réel l'état des serveurs, des applications et des réseaux, permettant ainsi de détecter et de résoudre rapidement les incidents avant qu'ils n'impactent les utilisateurs finaux. En outre, elle fournit des analyses et des rapports détaillés qui aident à optimiser les ressources et à prévenir les problèmes futurs. Cela garantit non seulement une meilleure satisfaction des clients grâce à un service fiable et performant, mais aussi une réduction des coûts liés aux temps d'arrêts et aux défaillances, renforçant ainsi la compétitivité et la réputation de Soldigit sur le marché.

Pour le choix de l'outil, il nous faut définir des critères nous permettant d'évaluer les différents outils recensés.Le tableau suivant présente donc ces critères, leurs définitions (explications) ainsi que les méthodes d'évaluation qui seront utilisées.

TABLE 2.1 : Grille comparative des solutions de supervision.

Caractéristiques	Splunk Entreprise	QRadar	Nagios Core	Stack ELK
Expérience Utilisateur	Excellente	Très Bonne	Bonne	Excellente
Règles et actions personnalisables	Personnalisation avancée pour les grandes entreprises	Personnalisation avancée pour les grandes entreprises	Hautement personnalisable	Personnalisation avec interface conviviale
Côut	Variété d'options tarifaires	Élevé pour les licences et la maintenance	Open-source, options commerciales	Open-source, options commerciales
Protection des données	Très Elevée	Très Elevé	Moyenne	Elevée
Grande communauté d'utilisateur	Restreinte	Restreinte	Large	Large
Gestion des alertes	Excellente	Très Bonne	Bonne	Très Bonne

2.2 Fonctionnement du stack ELK

Le stack ELK (Elasticsearch, Logstash et Kibana) est un ensemble de trois outils open source. En exploitant le format JSON, il offre une analyse automatique en temps réel, un algorithme de recherche très flexible et différents types de visualisations (histogrammes, graphiques, diagrammes circulaires, etc.) pour chaque donnée. La figure ci-dessous illustre son modèle de fonctionnement.

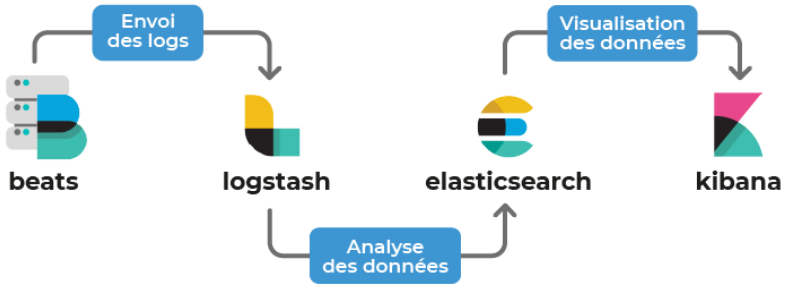


FIGURE 2.1 : Fonctionnement du Stack ELK.[4]

2.2.1 Elasticsearch[5]

Elasticsearch est un moteur de recherche OpenSource utilisant Lucene pour l'indexation. Il sert en quelque sorte de base de données NoSQL dans le stack ELK. Elasticsearch est livré avec les fichiers de configuration suivants :

- **elasticsearch.yml :** Il est utilisé pour configurer les différents modules Elasticsearch.
- **logging.yml :** Ce fichier est utilisé pour configurer la journalisation d'Elasticsearch.

Ces deux fichiers de configuration se trouvent dans le dossier ES_HOME/config.

Il est un outil extrêmement complet et une description exhaustive de toutes ses fonctionnalités dépasse largement le cadre de notre projet de recherche. Mais, retenons que le format de configuration par défaut de Elasticsearch est YAML avec pour port 9200. Ce fichier au format YML dispose de plusieurs options à savoir :

- **cluster.name :** Permet la jointure de tous les nœuds ;
- **node.name :** Permet d'identifier le nœud (Exemple : **$HOSTNAME**) ;
- **node.master :** Détermine si le nœud gère le cluster ou pas ;
- **node.data :** Détermine si le nœud héberge la donnée ou pas ;
- **path.data :** Définit où est stockée la donnée sur le nœud ;
- **path.log :** Définit où sont stockés les logs sur le nœud ;
- **http.port :** Définit le port d'écoute d'Elasticsearch (9200 par défaut).

2.2.2 Logstash[1]

Logstash est un moteur de collecte et de traitement des données via plugin. Il est doté de nombreux plugins qui permettent de configurer facilement l'outil pour collecter et filtrer les données et les mettre en forme pour l'injecter dans elastisearch. Le fichier de configuration par défaut de logstash se retrouve dans le répertoire **/etc/logstash/logstash.yml**. Il est constitué de trois phases l'Input, le Filter et l'Output.
L'**Input** indique la manière dont les données seront entrées dans le pipeline. En regardant, la documentation du **plugin d'entré file**, nous remarquons que nous pouvons utiliser différents paramètres pour cette entrée. Comme par exemple, le paramètre **start position** qui permet de choisir où Logstash doit commencer sa lecture initiale des fichiers, soit au début ou à la fin. La valeur par défaut est **end** qui traite les fichiers comme des flux directs en temps réel (comme la commande tail -f) et comme son nom l'indique, la lecture débutera à partir de la fin du fichier, mais si nous souhaitons lire un fichier depuis le début, il suffit de forcer alors la valeur à beginning :

```
input {
    file {
        path => "/var/log/apache2/access.log"
        start_position => "beginning"
        sincedb_path => "/dev/null"
    }
}
```

FIGURE 2.2 : Configuration de la **phase Input**

Le **Filter** est le module pour traiter les données récupérées depuis l'input. Au cours de ce travail, nous utiliserons le plugin **grok** qui nous permettra d'analyser facilement les données de logs non structurées en quelque chose de plus exploitable en utilisant des expressions régulières. La syntaxe d'un pattern grok est la suivante :

```
%{SYNTAX:SEMANTIC}
```

FIGURE 2.3 : Syntaxe d'un pattern grok

- **SYNTAX** : nom du pattern qui correspond à un filtre prédéfini, comme par exemple le pattern NUMBER qui prend en compte des nombres tel quel 3 ou 3.55 ou le pattern IP qui correspond à une IP comme 192.168.1.20.
- **SEMANTIC** : est l'identifiant de la valeur récupérée par votre SYNTAX.

Le point d'arrivée de toute configuration Logstash gérer avec l'**Output**, utilise différents plugins de sortie qui envoient les données traitées par la phase de filtrage à une destination particulière (ex : elasticsearch). Par ailleurs, les sorties sont la dernière étape du pipeline Logstash. Ici, **nous récupérons les évènements sur la sortie standard** avec le plugin de sortie stdout avant de les envoyer plus tard à elasticsearch. Le plugin **stdout** propose différents formats de sortie. Dans notre exemple, nous utiliserons le format rubydebug, comme ceci :

```
output {
    stdout { codec => rubydebug }
}
```

FIGURE 2.4 : Configuration de la **phase Output**

2.2.3 Kibana

Spécialisé dans la visualisation des données et la mise en place des dashboards, **Kibana est un client pour ElasticSearch qui fournit à l'utilisateur une interface graphique (UI) accessible par un navigateur web**. Cette UI permet à l'utilisateur de réaliser des recherches, de visualiser des documents individuels et d'agréger des résultats dans des graphes et des diagrammes permettant ainsi aux utilisateurs de transformer rapidement d'énormes volumes de données brutes en informations exploitables.

2.2.3.1 Caractéristiques de Kibana

Nombreuses et variées sont les caractéristiques de Kibana et la plupart de ces caractéristiques montrent la particularité de Kibana parmi les plateformes de visualisation et d'analyse de données. Parmi elles, nous pouvons énumérer :

- Création de rapports Kibana : tableaux, graphiques, géographiques ;
- Création de tableaux de bord ;
- Exposition d'éléments de reporting avec un chemin d'accès sous forme d'URL ;
- Rafraichissement instantané ;
- Travail sur des données temps réel ;
- Possibilité de créer agrégations et filtres ;
- Réaliser des extractions.

2.2.3.2 Interface web de Kibana

Kibana dispose d'une interface web simple, interactive et facile à prendre en main, permettant la visualisation en permanence des données de supervision. Le fichier de configuration par défaut de Kibana se retrouve dans le répertoire **etc/kibana/kibana.yml** avec pour port **5601**. Si des modifications sont apportées à ce fichier, il est important que la configuration Kibana possède les bonnes informations pour communiquer avec Elasticsearch. Dans notre cas, nous tapons dans notre navigateur **http ://localhost :5601** après installation.

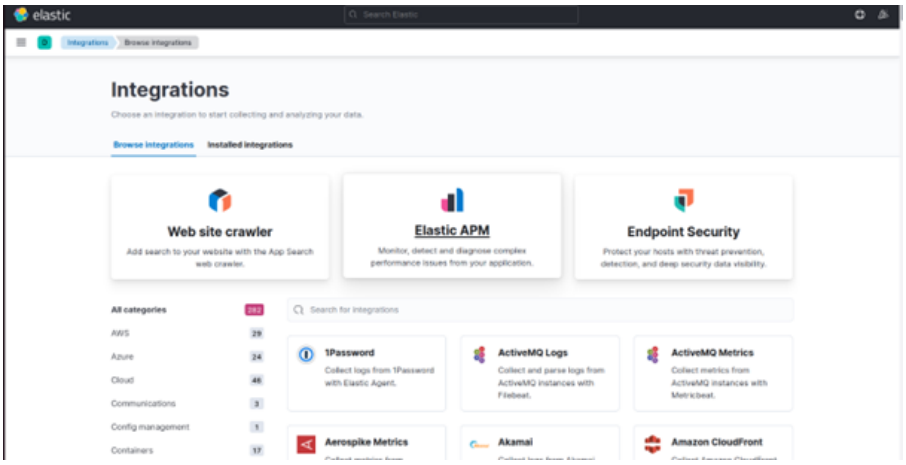


FIGURE 2.5 : Présentation de l'interface web de Kibana.

L'interface de Kibana se compose de quatre éléments principaux : **Discover, Visualize, Dashboard** et **Settings**.

1. **Discover**

   Kibana fournit une page Discover pour explorer les données correspondantes selon le modèle d'index sélectionné. Sur cette page, nous pouvons interroger les données, effectuer des recherches à l'aide de filtres et afficher les données d'un document. Nous pouvons également voir le nombre de résultats correspondants et les statistiques relatives à un champ.
   Si nous configurons le champ d'horodatage dans les données indexées, il affichera également, par défaut, un histogramme montrant la distribution des documents dans le temps.

2. **Visualize**

   Kibana propose la page Visualize pour créer de nouvelles visualisations. Celles-ci peuvent être basées sur :

   - la base de différentes sources de données ;
   - une nouvelle recherche ou une visualisation sauvegardée.

   Dans le cas où nous avons des données brutes d'une agrégation composée, nous pouvons les exploiter sous plusieurs formes selon nos préférences.

   - Graphique linéaire : Permet de comparer facilement différentes séries.
   - Widget Markdown : Permet d'afficher des informations ou des instructions sur notre tableau de bord.
   - Métrique : Idéal pour afficher un seul chiffre sur votre tableau de bord.

- Diagramme circulaire : Facilite la visualisation de la contribution de chaque source à un total.
- Carte en mosaïque : Permet de relier les résultats d'une agrégation avec des points géographiques.
- Diagramme à barres verticales : Il s'agit d'un diagramme à usage général.

Ces visualisations peuvent être sauvegardées, utilisées individuellement ou dans des tableaux de bord.

3. **Dashboard**

   Le tableau de bord est une collection de visualisations enregistrées dans différents groupes. Nous pouvons organiser ces visualisations librement grâce à une fonction de glisser-déposer. Elles peuvent être ordonnées en fonction de l'importance des données. Les tableaux de bord peuvent être facilement sauvegardés, partagés et chargés ultérieurement.

4. **Settings**

   Nous pouvons utiliser la page Paramètres pour configurer les indices Elasticsearch que vous souhaitez explorer. Nous sommes aussi en mesure de spécifier divers modèles d'index. Cette page montre les différents champs indexés dans un modèle d'index et les types de données de ces champs. On peut également créer des champs scriptés, qui sont calculés à la volée à partir des données.

2.3 Les agents Beats

Les agents Beats représentent des **Data shippers** qui se composent d'une bibliothèque principale : **libbeat**. Elle fournit une API pour expédier les données depuis la source, configurer les options d'entrées et implémenter la journalisation. Ces agents sont installés au niveau de la machine cliente. Il existe plusieurs types de beats à savoirs :

2.3.1 FileBeat

Filebeat est un agent de transfert léger ce qui nécessite moins de ressource. Il facilite l'envoie des données vers le centre de stockage où elles sont les plus utiles. Le fichier de configuration par défaut de filebeat se retrouve dans le répertoire **/etc/filebeat/filebeat.yml**. C'est également un constituant du stack, ce qui signifie qu'il fonctionne avec Elasticsearch, Logstash et Kibana. C'est un choix parfait pour collecter des logs sur des machines distantes.

2.3.2 MetricBeat

Metricbeat est un agent léger qui peut être installé sur les serveurs cibles pour collecter périodiquement des métriques à partir des serveurs cibles. Il peut s'agir de métriques de système d'exploitation tels que le processeur ou la mémoire ou des données liées aux services exécutés sur le serveur.
Tout comme Filebeat, il peut être configuré pour envoyer directement la sortie à Elasticsearch ou à Logstash seulement si les données doivent subir une transformation au préalable. Son fichier de configuration par défaut se retrouve dans le répertoire **/etc/metricbeat/metricbeat.yml**.

2.3.3 PacketBeat

Packetbeat tout comme les agents Beats présentés ci-dessus est un outil de surveillance réseau développé par Elastic qui utilise la bibliothèque **libpcap**. Il fournit des métriques de surveillance réseau en temps réel sur du protocole HTTP, TLS, DNS et de nombreux autres protocoles réseaux. Ainsi grâce à cet outil nous pouvons détecter des niveaux inhabituels de trafic réseau et des caractéristiques de paquets inhabituels, identifier les sources et les destinations des paquets, rechercher des chaînes de données spécifiques dans les paquets et à créer un tableau de bord convivial avec des statistiques pertinentes prêtes à l'emploi. Son fichier de configuration par défaut se retrouve dans le répertoire **/etc/packetbeat/packetbeat.yml**.

Conclusion

Le Stack ELK est un outil de supervision qui nécessite du temps lors de sa configuration, mais facilement maîtrisable. Son fonctionnement admet des similitudes avec les autres systèmes de supervision et également des spécificités. Il permet une supervision complète du système d'information, mais peut être associé avec d'autres outils comme les agents beats, pour limiter la consommation en ressource d'une part et améliorer la collecte des données de façon plus particulière.

Chapitre 3

Test et Implémentation

Introduction

Dans ce chapitre, nous allons aborder la configuration de l'architecture de supervision, ainsi que les procédures de déploiement et de vérification.

3.1 Mise en place de la solution à Soldigit

3.1.1 Présentaton de l'architecture existante

Avant toute modification de l'infrastructure réseau, il est essentiel de se remémorer sa forme actuelle, ce qui permettra de mieux évaluer les effets de la solution de surveillance après sa mise en œuvre.

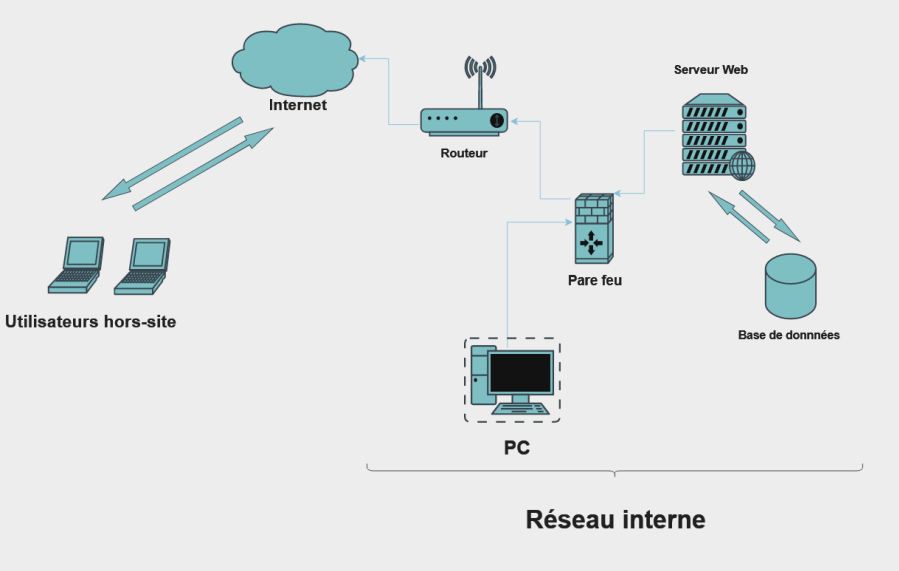


FIGURE 3.1 : Architecture de Soldigit

L'architecture de Soldigit repose sur un serveur web centralisé au sein de l'entreprise, desservant à la fois les utilisateurs internes et externes. Les utilisateurs internes se connectent directement au serveur via le réseau interne, assurant une communication rapide et sécurisée. Simultanément, les

utilisateurs externes accèdent au service web depuis Internet, protégés par un pare-feu qui limite les menaces potentielles. Cette configuration garantit une accessibilité optimale pour tous les utilisateurs, tout en maintenant un haut niveau de sécurité et de performance pour les opérations internes et externes.

3.1.2 Présentation de l'architecture implémentée

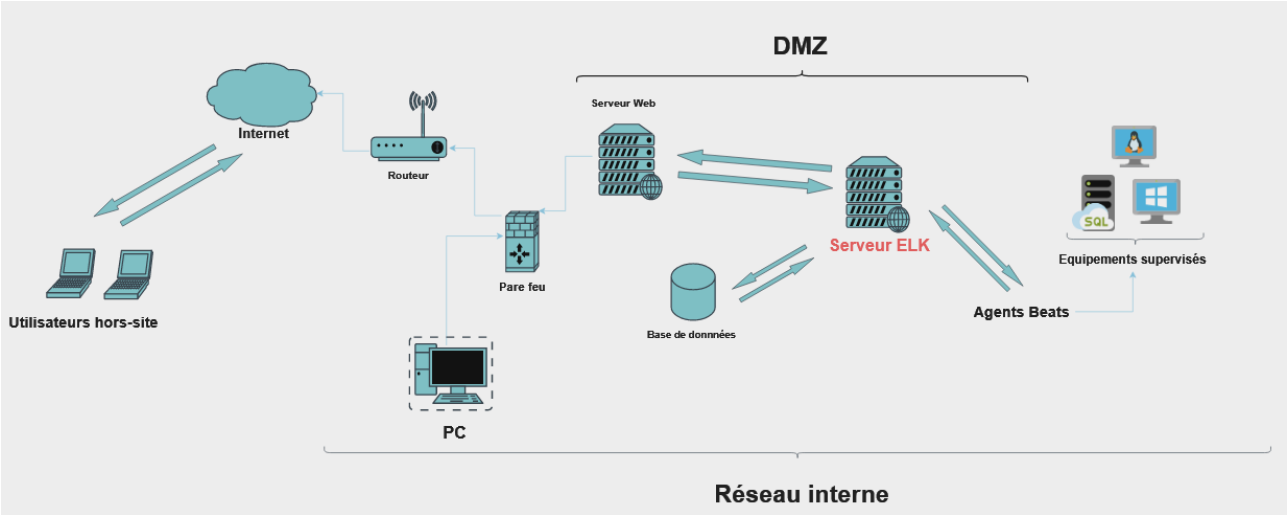


FIGURE 3.2 : Architecture de Soldigit après déploiement de la solution

Avec le déploiement de la solution au siens de Soldigit, l'architecture de l'entreprise a été considérablement renforcée pour garantir une surveillance continue et proactive de ses services web. La solution de supervision, basée sur le stack ELK permet une collecte, une analyse et une visualisation en temps réel des données provenant de divers composants de l'infrastructure. Les agents Beats sont déployés sur les machines clients pour collecter les logs et les métriques systèmes, qui envoient ces informations à Elasticsearch pour le traitement et le stockage. Kibana fournit une interface utilisateur intuitive pour visualiser les données et générer des rapports détaillés. Cette architecture permet à l'équipe de Soldigit de détecter rapidement les anomalies, d'identifier les goulots d'étranglement, et de réagir immédiatement aux incidents, assurant ainsi une haute disponibilité et une performance optimale de leurs services web pour leurs clients.

3.1.2.1 Illustration du mécanisme de fonctionnement de la solution proposée

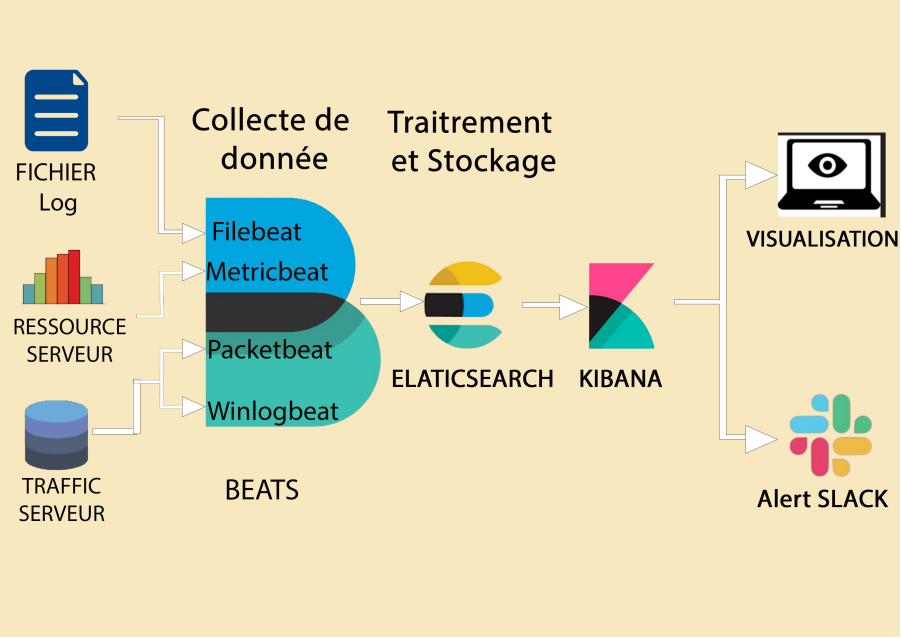


FIGURE 3.3 : Mécanisme de fonctionnement de la solution proposée

En premier lieu, la collecte de données est prise en charge par des agents Beats spécifiques. Filebeat s'occupe des fichiers de log, Metricbeat des ressources serveur, Packetbeat du trafic réseau au niveau des systèmes d'exploitation Linux, et Winlogbeat assure la même fonction pour le système d'exploitation Windows. Une fois ces données collectées, elles sont traitées et stockées dans Elasticsearch. Ensuite, les informations sont transférées vers Kibana pour la visualisation, et notre outil d'alerte Slack signale toute activité critique.

3.2 Installation et configuration de la solution

3.2.1 Machine serveur de supervision

Une fois connecté sur la machine considérée comme le serveur de supervision, nous sommes passés à l'installation d'Elasticsearch et Kibana. Mais, bien avant de lancer l'installation des outils côté serveur nous avons eu à faire la vérification des pré-requis pour un parfait fonctionnement de l'environnement.

1- Phase d'Installation

Tout d'abord, nous avons ajouté la clé de signature d'Elastic pour que le package téléchargé puisse être vérifié en utilisant la commande :

$ `wget-qO-https://artifacts.elastic.co/GPG-KEYelasticsearch|sudo apt-keyadd`

Nous sommes ensuite passés à l'installation du paquet apt-transport-https sur notre machine Debian avec la commande **sudo apt-get install apt-transport-https** afin de garantir une intégrité des paquets téléchargés et se protéger contre les attaques potentielles telles que l'interception de paquets non chiffrés.

L'étape suivante consiste à ajouter le dépôt Elasticsearch sur le système debian.

$ `echo"debhttps://artifacts.elastic.co/packages/7.x/aptstablemain"|sudo tee-a/etc/apt/sources.list.d/elastic-7.x.list`

- Une fois toutes ses commandes exécutées, nous avons fait une mise à jour de nos références puis installé convenablement nos outils.

$ **sudo apt-get update -y && sudo apt-get install elasticsearch**

Après installation nous exécutons Elasticsearch avec la commande :

$ **sudo systemctl start elasticsearch**



FIGURE 3.4 : Interface Web Elasticsearch après installation

Pour confirmer le bon fonctionnement, nous avons saisie `http://192.168.1.197:9200` dans notre navigateur.

Pour initialiser le service à chaque démarrage de la machine, lançons la commande suivante :

$ **sudo systemctl enable elasticsearch**

- Après la mise en service d'Elasticsearch, nous sommes passés à celui de Kibana tout en maintenant les pré-requis qui ne sont plus obligatoire. Alors exécutons la commande d'installation et de démarrage du service :

$ **sudo apt-get install kibana && sudo systemctl start kibana**

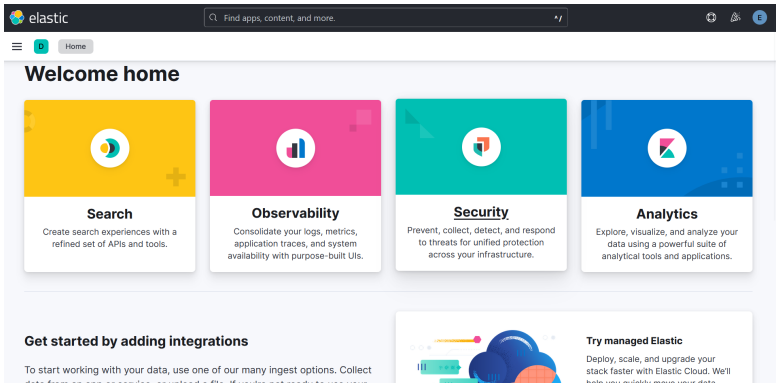


FIGURE 3.5 : Interface web kibana après installation

Pour confirmer le bon fonctionnement, nous avons saisi `http://192.168.1.197:5601` dans notre navigateur.

2- Phase de Configuration

Nous allons nous intéresser au répertoire de configuration des différents services installés afin de rendre la solution de supervision exploitable pour notre projet.

- Pour le **service Elasticsearch** nous devons nous rendre dans le répertoire **/etc/elasticsearch/ elasticsearch.yml** à ce niveau définir les paramètres suivants comme l'indique la figure ci-dessous :

```
  GNU nano 6.2                          elasticsearch.yml *
#  ------------------------------------------------------------

path.data: /var/lib/elasticsearch
#
# Path to log files:
#
path.logs: /var/log/elasticsearch

network.host: "http://192.168.1.197"

# Enable security features
xpack.security.enabled: true

xpack.security.enrollment.enabled: false

# Enable encryption for HTTP API client connections, such as K
xpack.security.http.ssl:
```

FIGURE 3.6 : Extrait paramètre de configuration elasticsearch

- Le **cluster.name :** qui définit le nom du cluster Elasticsearch est mis sur ES et est généralement par défaut exploité par l'application ;

- Le **node.name : node-1** spécifie le nom du nœud Elasticsearch. Chaque nœud doit avoir un nom unique au sein du cluster ;

- Précision du **port http** qui est dans notre cas le **9200** l'une des configurations importantes pour la communication entre les services ;

- **cluster.initial_master_nodes : ["nœud1"],** il s'agit des nœuds qui participent à l'élection du maître lors de la formation initiale du cluster ;

-**xpack.security.enabled** permet d'activer ou désactiver la sécurité X-Pack.

Au niveau de la sécurité elasticsearch, nous sommes passés en **mode super utilisateur** ou **mode root** au répertoire**/usr/share/elasticsearch/bin**. Ensuite, nous avons exécuté la commande :

$ **./elasticsearch-setup-passwords interactive**

Grâce à cette commande, nous avons créé des mots de passe pour les utilisateurs intégrés, leur permettant de communiquer de manière sécurisée avec le cluster.

- Pour le **service Kibana**, la configuration semble un peu plus complexe. Une fois les configurations de bases achevées, nous devons générer des keystores ainsi que les passwords afin de garantir une communication entre les services sur le même noeud tout en maintenant la sécurité de notre solution de supervision. En plus du répertoire habituel **cd /etc/kibana/kibana.yml** pour la configuration, nous nous sommes rendus dans le répertoire **/usr/share/kibana/bin**.

À ce niveau, il fallait configurer Kibana pour qu'il se connecte à Elasticsearch à l'aide d'un mot de passe. Avec la commande **./Kibana-keystore-create** nous sommes parvenu à créer une clé de connexion puis nous l'avons ajouté au répertoire de stockage des mots de passe d'Elasticsearch à l'aide de la commande **./Kibana-keystore-create add elasticsearch Password.**

```
 GNU nano 6.2                          kibana.yml *
# ================== System: Elasticsearch ==================
# The URLs of the Elasticsearch instances to use for all your queries.
server.host: "192.168.1.197"

elasticsearch.hosts: "http://192.168.1.197:9200"

# If your Elasticsearch is protected with basic authentication, these settings provide
# the username and password that the Kibana server uses to perform maintenance on the>
# index at startup. Your Kibana users still need to authenticate with Elasticsearch, >
# is proxied through the Kibana server.
elasticsearch.username: "kibana_system"
elasticsearch.password: "elastic"

xpack.encryptedSavedObjects.encryptionKey: 6bb98a38a95ac4c13baac0ac77b1bc02
xpack.reporting.encryptionKey: bc36ce2ed3e4b438515561be489a5e6d
xpack.security.encryptionKey: 33320d7334f3c5e834f507a8d9435401
```

FIGURE 3.7 : Répertoire de Configuration kibana

La capture ci-dessus présente les données de connexion intégrées ainsi que l'ajout de la clé de connexion.

3.2.2 Machine client

3.2.2.1 Cas d'une machine client ubuntu

Avant de commencer l'installation, nous nous sommes assurés d'avoir installé Elasticsearch pour stocker et rechercher nos données, et Kibana pour les visualiser et les gérer. Après la vérification, nous avons fait une mise à jour de notre système ainsi que des paquets avec la commande :
$ **sudo apt update -y && sudo apt upgrade -y**

1- Phase d'Installation

Comme pré-requis nous avons installé le paquet **apt-transport-https** et la **clé de signature publique** respectivement :

$ **sudo apt-get install apt-transport-https**

$ `wget-qO-https://artifacts.elastic.co/GPG-KEY-elasticsearch|sudo apt-keyadd-`

L'étape suivante consiste à ajouter le dépôt Elastic sur notre système afin de démarrer l'installation proprement dite des agents beats :

$ `echo"debhttps://artifacts.elastic.co/packages/7.x/apt stablemain"|sudo tee-a/etc`

$ `/apt/sources.list.d/elastic-7.x.list`

- Pour le **service filebeat**, nous avons mis à jours nos référentiels puis installer Filebeat :

$ **sudo apt-get update && sudo apt-get install filebeat**

Une fois l'installation faite nous avons démarré le service avec **sudo systemctl start filebeat**.

```
● filebeat.service - Filebeat sends log files to Logstash or directly to Elas
     Loaded: loaded (/lib/systemd/system/filebeat.service; enabled; vendor pr
     Active: active (running) since Sat 2024-06-08 09:17:21 WAT; 1min 18s ago
       Docs: https://www.elastic.co/beats/filebeat
```

FIGURE 3.8 : Statut actif sur le service filebeat

La capture ci-dessus indique que l'agent Filebeat est actuellement actif.

- Pour le **service Metricbeat**, nous sommes passés directement à l'installation et au démarrage du service :

$ **sudo apt-get install metricbeat && sudo systemctl start metricbeat**

```
● metricbeat.service - Metricbeat is a lightweight shipper for metrics.
     Loaded: loaded (/lib/systemd/system/metricbeat.service; enabled; ve
     Active: active (running) since Sat 2024-06-08 09:18:52 WAT; 56s ago
       Docs: https://www.elastic.co/beats/metricbeat
```

FIGURE 3.9 : Statut actif sur le service metricbeat

La capture ci-dessus indique que l'agent Metricbeat est actuellement actif.

- Pour le **service Packetbeat**, nous sommes également passés à l'installation et au démarrage du service :

$ **sudo apt-get install packetbeat && sudo systemctl start packetbeat**

```
● packetbeat.service - Packetbeat analyzes network traffic and sends the
     Loaded: loaded (/lib/systemd/system/packetbeat.service; enabled; ven
     Active: active (running) since Sat 2024-06-08 09:14:15 WAT; 6min ago
       Docs: https://www.elastic.co/beats/packetbeat
```

FIGURE 3.10 : Statut actif sur le service packetbeat

La capture ci-dessus indique que l'agent Packetbeat est actuellement actif.

2- Phase de Configuration

Pour la phase de configuration des agents beats, les paramétrages sont presque similaires. Nous nous sommes basés sur les modules de chaque agent beat afin de simplifier la configuration et la collecte de données spécifiques. En utilisant ces modules, nous pouvons configurer rapidement Beats pour collecter des données spécifiques à une application ou à un service sans avoir à définir manuellement tous les paramètres.

- Au niveau du **service filebeat**, nous avons ajouté les informations (identifiant et le mot de passe) de connexion à l'interface kibana. Ensuite, insérer l'adresse et le port de sortie d'Elasticsearch et Kibana. Et pour finir, activé le module pour la collecte de journaux Apache (module "apache2"). Tout ceci est fait dans le répertoire **/etc/filebeat/filebeat.yml**.

```
 GNU nano 6.2                          filebeat.yml *
# Configure what output to use when sending the data collected by the beat.

# ---------------------------- Elasticsearch Output ----------------------
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: "192.168.1.197:9200"

  # Performance preset - one of "balanced", "throughput", "scale",
  # "latency", or "custom".
  preset: balanced

  # Protocol - either `http` (default) or `https`.
  #protocol: "https"
  username: "elastic"
  password: "elastic"
```

FIGURE 3.11 : Fichier de configuration du service filebeat

- Au niveau du **service Metricbeat**, nous avons également procédé comme pour le fichier de configuration Filebeat. Ajoutez les informations de connexion (identifiant et mot de passe) à l'interface Kibana, insérez l'adresse et le port de sortie d'Elasticsearch et de Kibana. Pour finir, activez le module pour obtenir les statistiques sur l'utilisation du processeur, de la mémoire, du système de fichiers, sans oublier celles associées à chacun des processus exécutés sur notre système. La configuration se fait dans le répertoire **/etc/metricbeat/metricbeat.yml**.

```
  GNU nano 6.2                                  metricbeat.yml *
  index.codec: best_compression
  #_source.enabled: false
setup.dashboards.enabled: true

setup.kibana:

host: "192.168.1.197:5601"
# ------------------------------ Elasticsearch Output --
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: "192.168.1.197:9200"
  preset: balanced

  # Protocol - either `http` (default) or `https`.
  username: "elastic"
  password: "elastic"
```

FIGURE 3.12 : Fichier de configuration du service metricbeat

- Au niveau du **service packetbeat**, nous avons ajouté les informations (identifiant et le mot de passe) de connexion à l'interface kibana, inséré l'adresse et le port de sortie d'Elasticsearch et Kibana. Cette configuration se fait dans le répertoire **/etc/packetbeat/packetbeat.yml**.

```
  GNU nano 6.2                          packetbeat.yml
- type: dhcpv4
  # Configure the DHCP for IPv4 ports.
  ports: [67, 68]

- type: dns
  # Configure the ports where to listen for DNS traffic. You can disable
  # the DNS protocol by commenting out the list of ports.
  ports: [53]

- type: http
  # Configure the ports where to listen for HTTP traffic. You can disable
  # the HTTP protocol by commenting out the list of ports.
  ports: [80, 8080, 8000, 5000, 8002]

- type: memcache
  # Configure the ports where to listen for memcache traffic. You can disable
  # the Memcache protocol by commenting out the list of ports.
```

FIGURE 3.13 : Fichier de configuration du service packetbeat

3.2.2.2 Cas d'une machine client Windows

Pour la supervision de notre machine Windows, le processus d'intégration des services est largement différent à celle sur un système d'exploitation ubuntu. Nous nous sommes rendu dans notre navigateur pour télécharger notre fichier **winlogbeat.zip** afin de pouvoir permettre à notre serveur de supervision de pouvoir collecté les informations du client. Une fois le téléchargement terminé la

commande suivante **.\install-service-winlogbeat.ps1** exécutée dans le powershell en mode administrateur nous a permis son installation. La phase de configuration est similaire aux services installés sur notre système ubuntu.

```
106
107  # ================================== Outputs ===================================
108
109  # Configure what output to use when sending the data collected by the beat.
110
111  # ---------------------------- Elasticsearch Output ----------------------------
112  output.elasticsearch:
113    # Array of hosts to connect to.
114    hosts: "192.168.1.102:9200"
115
116    # Protocol - either `http` (default) or `https`.
117    #protocol: "https"
118
119    # Authentication credentials - either API key or username/password.
120    #api_key: "id:api_key"
121    username: "elastic"
122    password: "elastic"
123
124    # Pipeline to route events to security, sysmon, or powershell pipelines.
125    pipeline: "winlogbeat-%{[agent.version]}-routing"
```

FIGURE 3.14 : Insertion des données de connexion

Une fois toutes ses étapes achevées, nous pouvons charger le tableau de bord Kibana puis démarrer le service respectivement grâce aux commandes **.\winlogbeat.exe setup && Start-Service winlogbeat.**

3.3 Mise en situation et test

Il s'agit de faire la supervision de notre machine cliente avec pour adresse **192.168.1.110**. Cette supervision consistera à relever les données de connexion au serveur apache, sa disponibilité, l'état du processeur, des activités en cours, etc.

3.3.1 Vérification et interprétation des résultats

Pour démarrer avec la phase de test, nous nous sommes rendu sur notre interface Kibana. Pour ce fait, saisissons dans notre navigateur l'addresse de connexion **192.168.1.197**. A ce niveau, saisir le **username : "elastic"**et le **password : "elastic"** pour les informations de connexion :

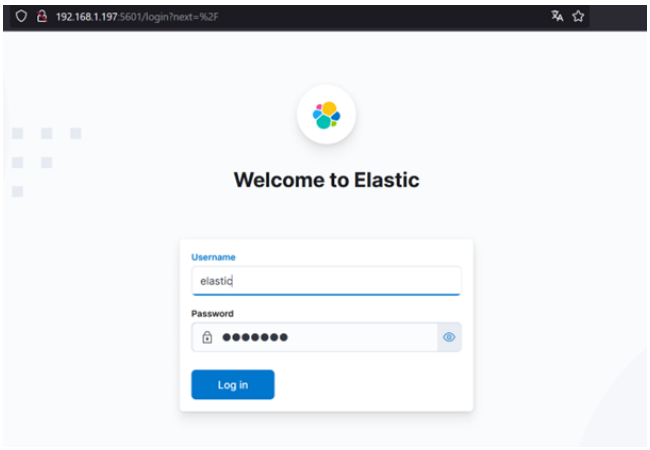


FIGURE 3.15 : Page d'authentification du stack ELK

3.3.1.1 Machine ubuntu

Une fois connectés, nous avons crée notre Index patterns afin de préciser l'agent beat sur lequel exploiter nos données. Une fois le choix effectué, nous pouvons créer le tableau de bord pour la visualisation.
Nous démarrons notre test avec **filebeat**. Pour mieux percevoir nos tests, nous avons lancé deux types de requêtes vers notre serveur local apache. Une, nous donnera accès au serveur puis l'autre qui est censé générer une erreur 404 vu que le répertoire d'url saisie est inexistant. Nous nous intéresserons à la requête présentant l'erreur d'accès au serveur local apache.

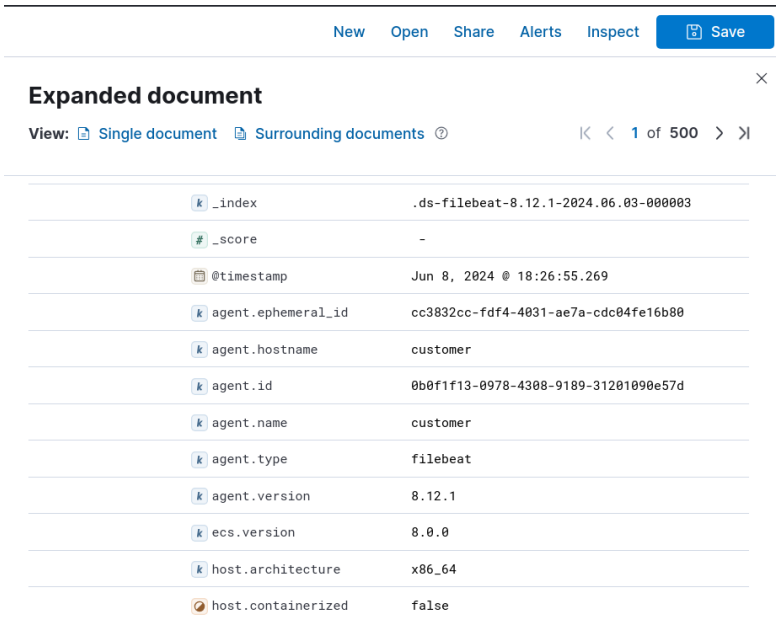


FIGURE 3.16 : Collecte d'information log avec filebeat

Afin de faire un traitement efficace des informations collectées, nous sommes passés au filtrage de certains paramètres spécifiques comme :

- Agent hostname (**customer)** : Ce paramètre est important dans l'identification du nom d'hôte afin d'avoir une idée sur la source des données.

- Service type (**Apache)** : Avec ce type de service, nous avons plus d'information sur notre collecte qui pourrait être exploiter dans le cas d'une potentielle attaque ou investigation.
- Input type (**log**) : En se basant sur le type d'entrée, nous avons la nature du fichier monitoré. Cela permet également de mieux orienter les filtres lors d'une recherche spécifique.
- log.file.path (**var/log/apache2/acces.log**) : Ce champ nous permet de connaître le chemin ou l'origine de la collecte effectuée sur notre machine.
- url.original (**test**) : Dans le cas de notre stimulation, nous avons utiliser comme url localhost/-test dans le but d'avoir un tel résultat. Ce qui confirme la valeur test trouvée par notre agent filebeat.
- traefik.acess.user_agent (**Firefox**) : Avec ce paramètre, nous avons la possibilité de connaitre le navigateur utilisé par le client lors de sa requête vers notre serveur apache.
- traefik.acess.user_agent.os_name (**Ubuntu**) : Nous permet d'avoir une idée sur le système d'exploitation utilisé par notre machine. Dans notre cas, nous sommes avec Ubuntu.

Une fois la mise en situation effectuée avec l'agent filebeat pour la supervision log, nous nous sommes intéressés aux métriques. En cliquant sur l'onglet overview de la section d'observation, nous avons la possibilité de sélectionner la machine (Host) là nous avons une vue d'ensemble de l'hôte à superviser.

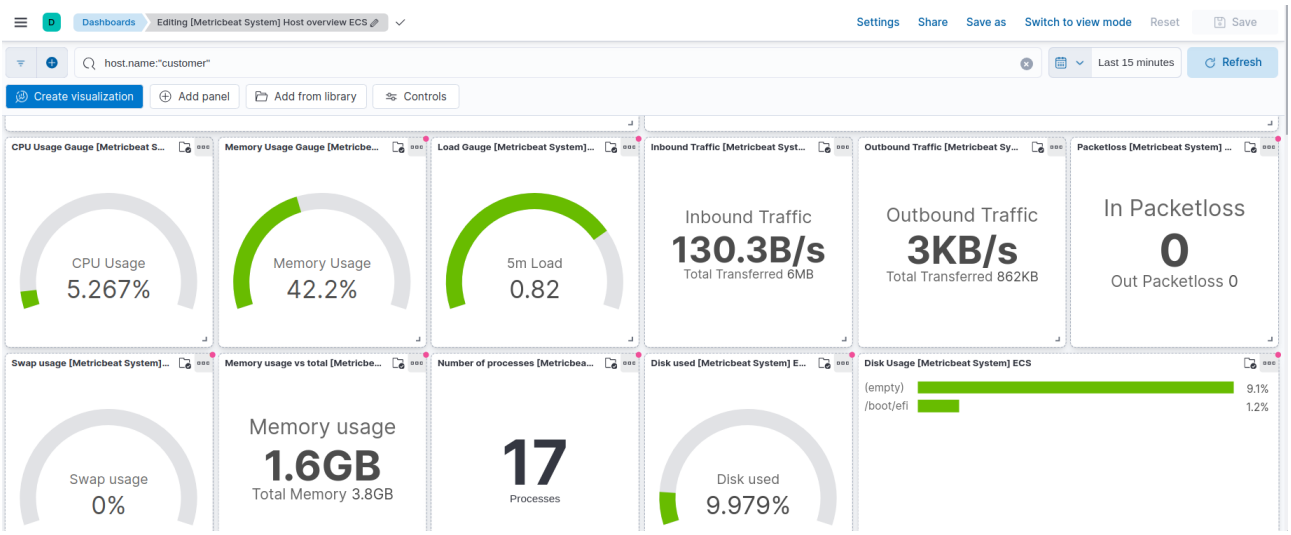


FIGURE 3.17 : Collecte d'informations avec l'agent metricbeat.

Les données collectées ci-dessus par l'agent Metricbeat fournissent une vue d'ensemble détaillée des performances et de l'utilisation des ressources du système. **L'utilisation du processeur** est relativement faible à **5,267%**, ce qui indique que le système n'est pas sous une charge de traitement intensive. **L'utilisation de la mémoire** est à **42,2%**, ce qui est raisonnable et laisse une marge pour les opérations supplémentaires. La **charge normalisée** de **0,82** est inférieure à **1**, ce qui signifie que le système n'est pas surchargé et fonctionne de manière optimale. Le trafic réseau montre une activité modérée avec un **trafic entrant** de **130,3B/s** et un **trafic sortant** de **3KB/s,** indiquant une utilisation stable du réseau sans signes de congestion.

La **mémoire RAM** utilisée est de **1,6GB**, ce qui, comparé à l'utilisation de la mémoire globale de **42,2%**, suggère que le système dispose de suffisamment de mémoire pour ses besoins actuels. Le

fait qu'il n'y ait aucune perte de paquets est un indicateur positif de la stabilité du réseau. Avec 17 processus en cours, le système semble gérer un nombre modéré d'applications et de services, et une utilisation de la **mémoire disque** de **9,979%** montre qu'il y a encore beaucoup d'espace de stockage disponible. Globalement, les métriques indiquent un système équilibré et performant, sans signes de goulets d'étranglement ou de surutilisation des ressources.

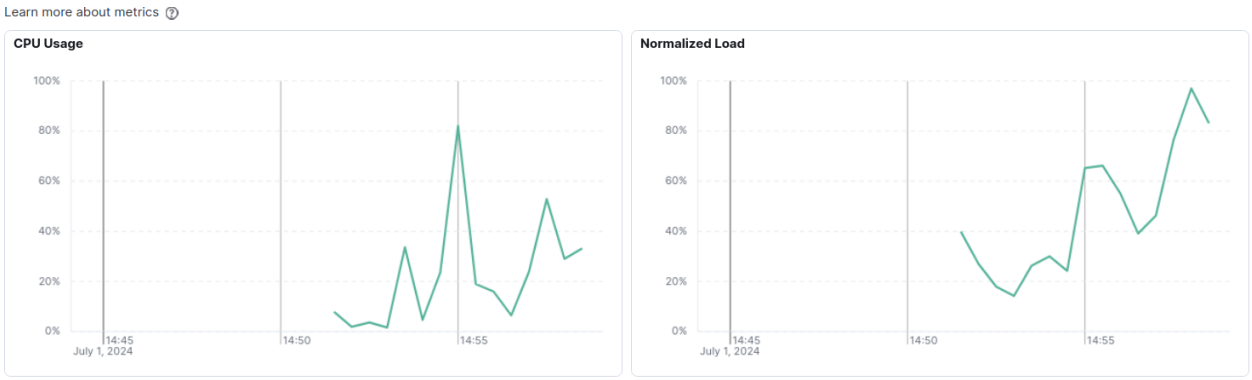


FIGURE 3.18 : Représentation des informations de collecte sous forme de graphe.

En se référant à la capture ci-dessus, le graphe de l'utilisation du processeur à gauche montre une variation significative des performances au fil du temps. À partir de 14h52, **l'utilisation du processeur augmente progressivement**, atteignant un **pic de 80%** à **14h55**. Ce pic peut être attribué aux diverses tâches exécutées lors du démarrage de la machine, telles que l'initialisation des services et les processus système. Cette utilisation élevée du processeur indique que ces tâches sont exigeantes en termes de ressources, ce qui peut ralentir d'autres opérations pendant cette période.

Après le pic, **l'utilisation du processeur chute rapidement à 20%**, puis **diminue progressivement jusqu'à 10%**. Cette décroissance rapide et stabilisation ultérieure montrent que le système gère efficacement les charges de travail intensives et retrouve un état stable sans maintenir une utilisation excessive du processeur. Cependant, un pic de 80% pourrait indiquer une opportunité d'optimisation pour réduire l'impact sur les performances globales. L'analyse critique révèle que, bien que le système soit capable de gérer des pics de charge et de revenir à une utilisation stable, des optimisations pourraient améliorer la réactivité et la stabilité pendant les périodes de forte sollicitation.

Toujours dans l'optique de faciliter la supervision, nous avons décidé de nous diriger vers un outil d'alerte pour nous tenir informés du moindre incident sur nos serveurs. C'est là qu'intervient **ElastAlert**[6], un cadre simple pour alerter sur les anormalies, les pics ou d'autres modèles d'intérêt des données dans le stack ELK. Cet outil nous permet d'écrire nos règles pour les alertes que nous souhaitons. Il s'agit d'un mécanisme d'alerte avec plusieurs types de sortie, tels que Slack, Email, JIRA, OpsGenie, etc.

Dans notre projet, nous nous sommes intéressés à la sortie **Slack**. Pour cela, après avoir créé le lien `https://hooks.slack.com/services/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` **webhook** sur notre canal, nous pouvons nous rendre sur notre interface de supervision afin de renseigner l'URL générée lors de la création de notre règle. Grâce aux données metricbeat, nous allons mettre en place une alerte qui nous informe sur l'utilisation de notre processeur.

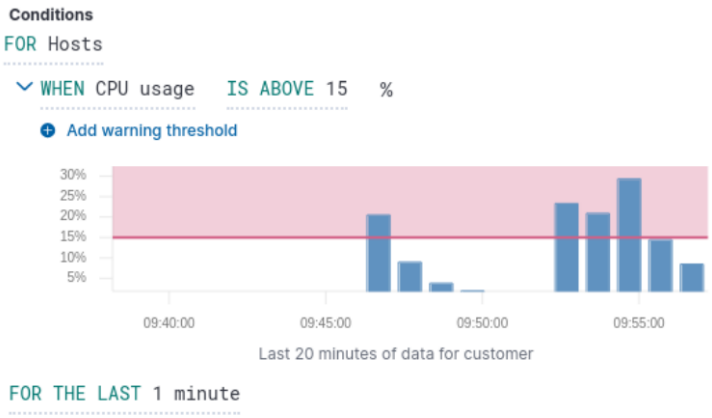


FIGURE 3.19 : Création de la règle avec metricbeat.

D'après l'image ci-dessus, une alerte est envoyée à l'administrateur système par notre outil Slack lorsque notre processeur atteint une valeur supérieure à 15% pour une durée d'une minute sur notre hôte.



FIGURE 3.20 : Réception du message d'alerte avec metricbeat.

La capture ci-dessus en constitue un exemple avec une valeur de 27,1% qui dépasse largement le seuil définit (27,1 > 15).

En ce qui concerne l'agent **packetbeat**, nous avons eu la possibilité de relever tous les trafics apparents sur notre système d'information.

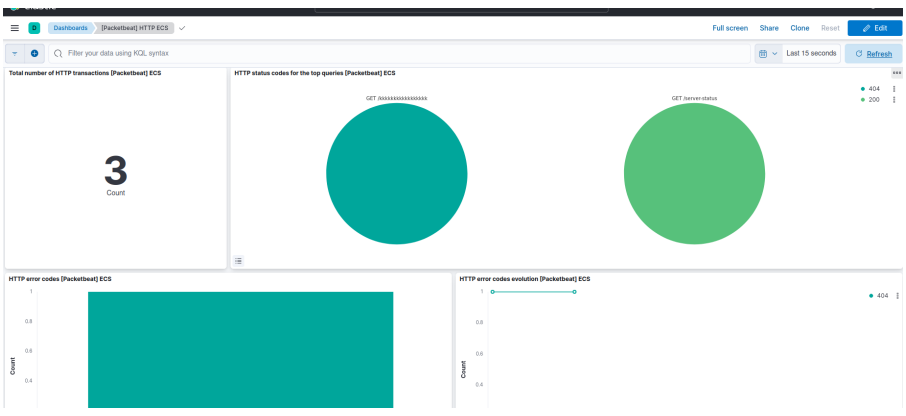


FIGURE 3.21 : Collecte d'informations avec packetbeat



FIGURE 3.22 : Suite d'informations de collecte packetbeat

Sur les captures ci-dessus nous observons le tableau de bord du traffic réseau sur une vue d'ensemble. On constate que :

- le **nombre de requête http** effectué est mentionné sur notre environnement de supervision. Celui-ci a une valeur égale à trois (**03**) ;
- **HTTP status codes for the top queries** : Ce paramètre indique l'URL ainsi que l'état de la requête. Ici les données montrent un échec de connexion avec pour **url : `localhost/test`** puis un état de succès avec pour url :`localhost`.

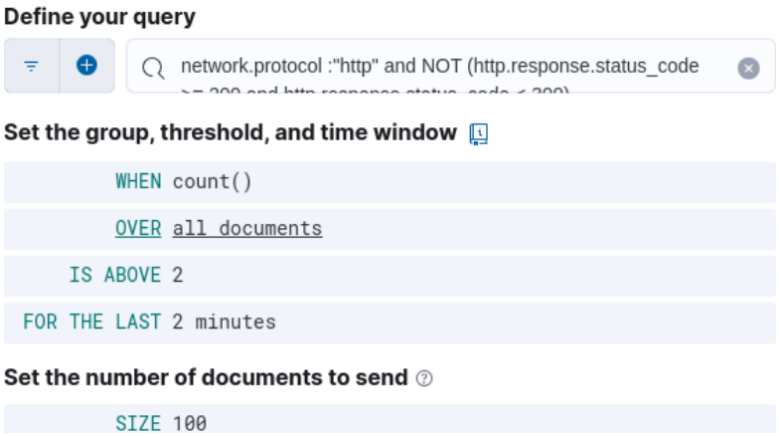


FIGURE 3.23 : Création de la règle avec packetbeat.

Pour rendre la supervision facile à l'administrateur système, nous avons configuré une règle qui alerte sur les **requêtes HTTP erronées**. Pour cela, nous avons défini la requête sur "**network.protocol :"http" and NOT (http.response.status_code >= 200 and http.response.status_code < 300)**". L'action est effectuée pour une requête supérieure à deux (02) avec une durée de deux minutes (02 min). Comme le présente la capture ci-dessus :

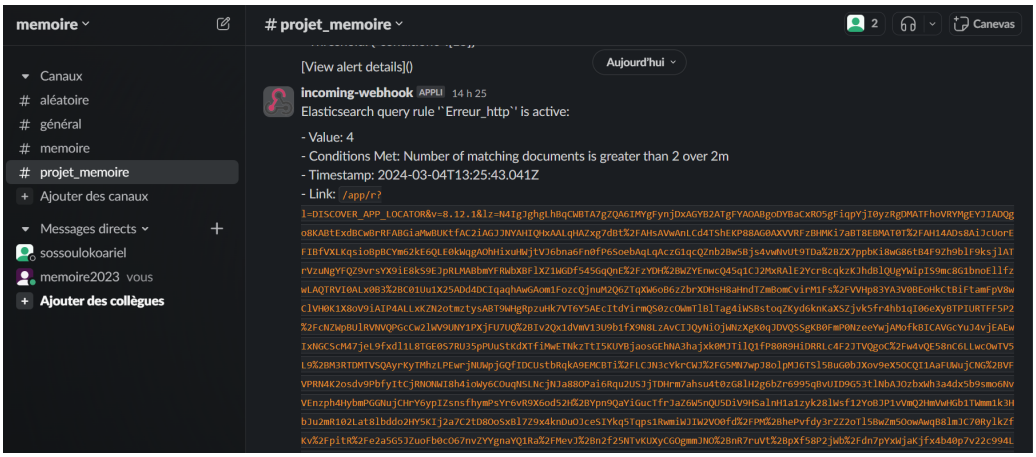


FIGURE 3.24 : Réception du message d'alerte avec packetbeat.

La capture ci-dessus confirme une bonne réception de l'alerte après avoir simulé notre requête.

3.3.1.2 Machine windows

Après l'affichage de notre interface de visualisation, nous nous sommes dirigés vers les dashbords.

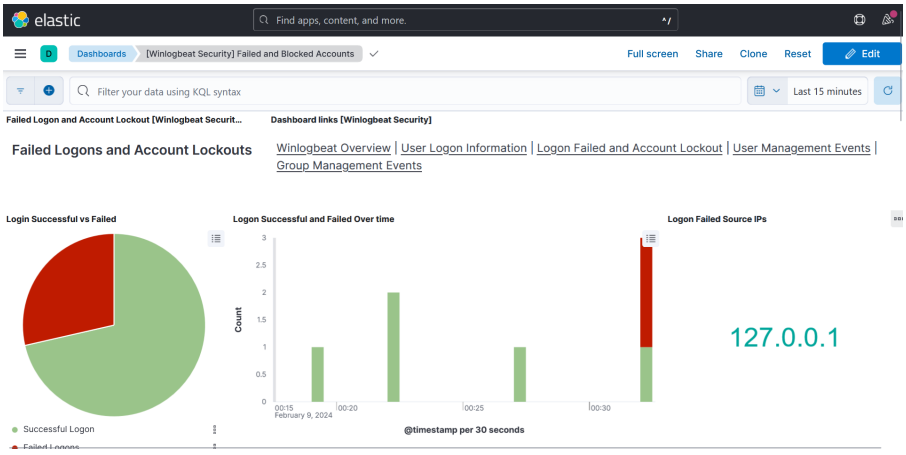


FIGURE 3.25 : Collecte d'information de connexion avec winlogbeat.

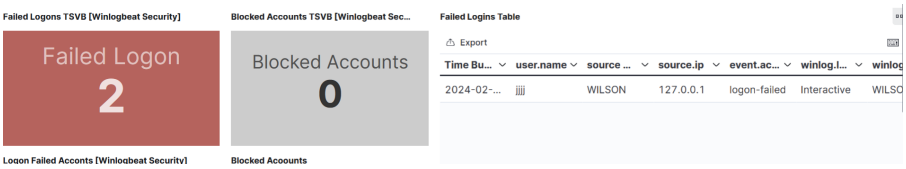


FIGURE 3.26 : Suite d'information de connexion.

Suite à l'analyse du **paramètre winlogbeat Security** représentée par la figure ci-dessus, nous notons qu'après avoir **simuler deux (02) mauvaises tentatives de connexion** à notre session, elle a été interprétée par notre outil de supervision par une colloration **rouge vif**. La couleur verte quant à elle indique un **accès réussi**. Soulignons que les collorations sont déterminées par nos choix personnels.

Conclusion

Dans cette section, nous avons dans un premier temps effectué la mise en place de la solution ELK à Soldigit. Nous avons par la suite configuré les différents serveurs et machines avant d'effectuer des tests de bon fonctionnement de cette solution.

Conclusion Générale

Dans ce mémoire, nous avons examiné le défi de l'indisponibilité des services web chez Soldigit, en recommandant l'implémentation d'un système de surveillance dans leur infrastructure informatique avec le stack ELK. Au cours du travail, nous avons d'abord fait un tour sur les protocoles et outils de supervision existants. Par la suite, une présentation détaillée du stack ELK puis la mise en place de l'architecture de supervision à Soldigit et présenté les différents résultats obtenus après des tests de validations.L'entreprise dispose donc d'une meilleure observabilité et peut répondre efficacement aux problèmes survenants sur les actifs monitorés.

La réalisation de ce projet nous a permis de mettre en œuvre nos connaissances acquises tout au long de notre formation et de mieux comprendre le fonctionnement d'une solution de supervision.

Conscient que notre travail est perfectible, nous proposons des axes de recherches en ce qui concerne les travaux futurs. Il s'agira de la mise en place d'un APM(Application Performance Monitoring) avec le stack ELK pour une surveillance complète de l'ensemble de l'infrastructure applicative. En parallèle, déployer un environnement de supervision spécifique pour le cloud, garantissant une surveillance constante des ressources et des services déployés dans cet environnement virtuel. Ces initiatives visent à assurer une gestion proactive et réactive des systèmes informatiques, facilitant l'identification rapide des problèmes et l'application de solutions efficaces pour maintenir la stabilité et l'efficacité opérationnelle.

Grâce à la mise en place de ce système, Soldigit est désormais mieux équipée pour identifier rapidement les problèmes et appliquer des solutions efficaces.

Bibliographie

[1] Applied ELK Stack : Actionable Data Insights and Business Metrics with the collective capability of ElasticSearch, Logstash and Kibana,Gurpreet S. Sachdeva,2017

[2] SUPERVISION IT SERVEURS, RÉSEAUX,APPLICATIONS ET ENVIRONNEMENTS VIRTUALISÉS, GUIDES COMPARATIFS,2023

Webographie

[3] https://world-connect.ch/services-informatique/monitoring/ consulté le 10/08/2023.

[4] https://images.app.goo.gl/2oDNGosmwdZiawtFA consulté le 10/08/2023. ‘

[5] https://www.next-decision.fr/editeurs-bi/restitution/elastic-kibana consulté le 22/08/2023.

[6] https://dev.to/morten12/security-alert-with-elastalert2-part--1b27 consulté le 01/16/2024.

[7] https://fr.wikipedia.org/wiki/Supervision_(informatique)#Liste_de_solutions_libres_et_open_source_de_supervisionconsulté le 22/08/2023.

[8] LeprotocoledesupervisionICMP(ProtocoledeMessagesdeContrôleInternet)\T1\textendashsupervisionréseaux:Solutiondesurveillancedeséquipementsconnectés(supervision-reseaux.fr) consulté le 02/10/2023.

[9] Lesavantagesincontournablesdelasupervisioninformatique(roverba.com) consulté le 05/10/2023.

[10] http://igm.univ-mlv.fr/~dr/XPOSE2010/supervision/nagios.html consulté le 10/10/2023.

[11] LogicielsdeSupervisionInformatique:Comparatif,Avis&Prix|Appvizer Consulté le 12/10/2023.

[12] outilspayantsdesupervision-Recherche(bing.com)consulté le 12/10/2023.

https://chat.openai.com consulté le 10/12/2023.

Table des matières

Dédicace **ii**

Remerciements **iii**

iii

Résumé **iv**

iv

Abstract **v**

v

Table des figures **vi**

Liste des tableaux **vii**

Liste des acronymes **viii**

Glossaire **1**

Introduction **2**

1 Revue de littérature **4**

Introduction . . . 4

1.1 Généralités sur la supervision . . . 4

1.1.1 Définition et objectif . . . 4

1.1.2 Importance de la supervision . . . 4

1.1.3 Fonctionnement de la supervision . . . 5

1.1.4 Protocoles de supervision . . . 5

1.1.5 Type de supervision[2] . . . 6

1.1.6 Les avantages de la supervision[9] . . . 7

1.2 Les différents types de solution de supervision . . . 7

1.3 Etude comparative des solutions propriétaires et open-source de supervision . . . 8

Conclusion . . . 8

2 Présentation du stack ELK **9**

Introduction . . . 9

2.1 Choix de la solution de supervision . . . 9

2.1.1 Justification et critère du choix pour l'utilisation d'une solution de supervision à Soldigit . . . 9

2.2 Fonctionnement du stack ELK . . . 10

2.2.1 Elasticsearch[5] . . . 10
2.2.2 Logstash[1] . . . 11
2.2.3 Kibana . . . 12
2.2.3.1 Caractéristiques de Kibana . . . 12
2.2.3.2 Interface web de Kibana . . . 13
2.3 Les agents Beats . . . 14
2.3.1 FileBeat . . . 14
2.3.2 MetricBeat . . . 14
2.3.3 PacketBeat . . . 15

3 Test et Implémentation **16**
Introduction . . . 16
3.1 Mise en place de la solution à Soldigit . . . 16
3.1.1 Présentaton de l'architecture existante . . . 16
3.1.2 Présentation de l'architecture implémentée . . . 17
3.1.2.1 Illustration du mécanisme de fonctionnement de la solution proposée 18
3.2 Installation et configuration de la solution . . . 18
3.2.1 Machine serveur de supervision . . . 18
3.2.2 Machine client . . . 21
3.2.2.1 **Cas d'une machine client ubuntu** . . . 21
3.2.2.2 Cas d'une machine client Windows . . . 23
3.3 Mise en situation et test . . . 24
3.3.1 Vérification et interprétation des résultats . . . 24
3.3.1.1 **Machine ubuntu** . . . 25
3.3.1.2 **Machine windows** . . . 30

Conclusion **32**

Bibliographie **33**

Webographie **34**

Table des matières **35**