6/18/2020

Présentation IDS

Ossec – Wazuh - Suricata

Hugo Charpentier
EFREI PARIS – ANS

Table des matières

Définitions

IDS

Un système de détection d'intrusion (IDS) est un dispositif ou une application logicielle qui surveille un réseau ou des systèmes pour déceler toute activité malveillante ou toute violation de politique de sécurité. Toute activité malveillante ou violation est généralement signalée à un administrateur ou est recueillie de façon centralisée au moyen d'un système de gestion des informations et des événements de sécurité (SIEM). (Cf : https://cisco.goffinet.org/ccnp/filtrage/concept-ids-ips/)

IPS

Les systèmes de prévention des intrusions (IPS), également connus sous le nom de systèmes de détection et de prévention des intrusions (IDPS), sont des dispositifs de sécurité réseau qui surveillent les activités du réseau ou du système pour détecter toute activité malveillante. Les principales fonctions des systèmes de prévention des intrusions sont d'identifier les activités malveillantes, d'enregistrer des informations sur ces activités, de les signaler et de tenter de les bloquer ou de les arrêter. (Cf : https://cisco.goffinet.org/ccnp/filtrage/concept-ids-ips/)

Le service Ossec

Introduction

Ossec est un système de détection d'intrusion basé sur l'hôte (HIDS), gratuit et open-source. Ce service a été créé en 2008 par Daniel B. Cid. Il permet :

- une analyse des logs de chaque agent rattaché au serveur Ossec
- une vérification de l'intégrité de fichiers
- une surveillance du registre Windows
- une détection des rootkits
- une génération d'alerte en temps réel
- la possibilité de déclencher des réponses actives (IPS)

Ossec est disponible sur un bon nombre de systèmes d'exploitation comme Linux, OS X, FreeBSD, Solaris, Windows…

Ce service possède un moteur d'analyse des logs qui est capable de corréler et d'analyser les logs provenant de plusieurs appareils et avec plusieurs formats.

Architecture

Ossec peut être mis en place de quatre manières :

- Mode serveur : Ce dernier va stocker les bases de données de vérification et d'intégrité des fichiers. Il reçoit les logs de chaque agent, les analyse et génère des alertes selon les résultats de l'analyse.
- Mode agent : Ce dernier va être installé sur les machines que l'on veut monitorer. Il collecte ses logs et les envoie au serveur Ossec pour une analyse approfondie.
- Mode hybride : Ce dernier va faire office à la fois de serveur et d'agent.
- Mode autonome : Ce dernier n'est rattaché à aucun serveur ou agent. Les décodeurs et les règles sont stockés localement.

La communication entre agent et serveur OSSEC est faite par défaut sur le port 1514 en UDP. Le port 1515 est utilisé pour l'échange de clé entre serveur et agent. Chaque communication est chiffrée par l'algorithme blowfish.

Principales caractéristiques

Contrôle de l'intégrité des fichiers

Chaque attaque contre une machine ou un réseau modifie les systèmes d'une manière ou d'une autre. L'objectif du « File Integrity Monitoring » (FIM) est de détecter ces changements et d'alerter lorsqu'ils se produisent. À la suite d'une attaque, d'un abus par un employé ou bien même d'une faute de frappe par un administrateur, tout changement de fichier, de répertoire ou de registre peut être signalé (suivant la configuration du service). Ossec permet de déclencher ce contrôle à des intervalles de temps réguliers et va comparer les hashs (MD5, SHA1) de chaque fichier avec celui qui a été enregistré précédemment.

Inspection des logs

Chaque système d'exploitation, application et appareil d'un réseau génère des logs. Ossec collecte, analyse et met en corrélation ces logs pour permettre de savoir si quelque chose de suspect se produit. Ossec produit des alertes selon des règles précises. Il est possible de créer/modifier/supprimer des règles selon le résultat d'une analyse de risque. Le service Ossec va analyser tous les logs qui lui parviennent en testant chaque règle présente. Si une règle, qui a un niveau d'alerte supérieur au seuil

d'alerte configuré par l'administrateur, est vérifiée, une alerte est produite. Les règles peuvent suivre une hiérarchie, si une règle est vérifiée, le service Ossec va tester toutes ses descendantes.

Détection de rootkit

Grâce à la détection de rootkit d'Ossec, ce dernier génère une alerte lorsque le système est modifié d'une manière commune aux rootkits. La méthode utilisée par Ossec est disponible à cette adresse : https://www.ossec.net/docs/manual/rootcheck/manual-rootcheck.html.

La détection de rootkit est réalisée selon différentes étapes :

- Par la lecture d'un fichier (« */var/ossec/etc/shared/rootkit_files* ») qui contient une base de rootkits et de fichiers couramment utilisés par ces derniers. Des appels systèmes (*fopen*, *opendir*…) vont être produit sur chacun des fichiers spécifiés.
- Par la lecture d'un autre fichier (« */var/ossec/etc/shared/rootkit_trojans* ») qui contient une base de données des signatures de fichiers malicieux. Chaque signature va être comparée à celle des fichiers spécifiés.
- Par l'analyse de l'ensemble du système à la recherche de fichiers inhabituels et de problèmes de permission par exemple les fichiers appartenant à l'utilisateur root avec une permission d'écriture à d'autres groupes d'utilisateur, les fichiers *suid* … Les répertoires et les fichiers cachés sont également inspectés.
- Par la recherche de processus cachés, les fonctions systèmes *getsid()* et *kill()* sont utilisés pour vérifier si un *pid* est utilisé ou non. Si un *pid* est utilisé mais que la commande *ps* ne fait pas référence à ce processus, cela indique la présence d'un rootkit au niveau du kernel ou bien une version piratée de la commande ps.
- Par la recherche de ports cachés, la fonction bind() est utilisé pour savoir si un port est ouvert ou non. Si le résultat de la fonction bind() est négatif (le port est utilisé) mais que la commande *netstat* ne spécifie pas que le port est utilisé, cela indique la présence d'un rootkit.
- Par l'analyse de chaque interface réseau. Si une interface réseau est en mode « promiscuous », alors le résultat de la commande *ifconfig* ou *ip* devrait y faire référence, sinon cela indique la présence d'un rootkit.

Réponse active

La réponse active d'Ossec permet d'exécuter des scripts lorsque des alertes spécifiques sont déclenchées. Il est possible de créer/modifier/supprimer ces scripts.

Le service Wazuh

Introduction

Wazuh est un système de détection d'intrusion basé sur l'hôte (HIDS), gratuit et open-source. C'est d'ailleurs un fork d'Ossec. De la même manière qu'Ossec, Wazuh permet principalement :

- une analyse des logs de chaque agent rattaché au serveur Ossec
- une vérification de l'intégrité de fichiers
- une surveillance du registre Windows
- une détection des rootkits
- une génération d'alerte en temps réel
- la possibilité de déclencher des réponses actives (IPS)

Architecture

Wazuh est du type agent-serveur tout comme Ossec :

- Mode serveur : Ce dernier va stocker les bases de données de vérification et d'intégrité des fichiers. Wazuh reçoit les logs de chaque agent, les analyse et génère des alertes selon les résultats de l'analyse.
- Mode agent : Ce dernier va être installé sur les machines que l'on veut monitorer. Il collecte ses logs et les envoie au serveur Wazuh pour une analyse approfondie.

La communication entre agent et serveur Wazuh est fait par défaut sur le port 1514 en UDP. Le port 1515 est utilisé pour l'échange de clé entre serveur et agent. Chaque communication est chiffrée par l'algorithme AES contrairement à blowfish pour Ossec.

Les différences avec Ossec

Wazuh détient toutes les caractéristiques d'Ossec et possède des améliorations notables (tous les changements et améliorations de Wazuh sont disponibles sur le github officiel de Wazuh.)

Voici les plus notables :

Évolutivité et fiabilité

- Dispositif anti-flooding pour éviter la perte d'un grand nombre d'événements ou un impact négatif sur les performances réseau.
- Support multi-thread pour les processus du serveur Wazuh, ce qui augmente considérablement les performances.

Compatibilité et intégration

- La compatibilité avec la suite ELK est accrue avec un plug-in Wazuh disponible. Ce dernier est aussi présent sur Splunk. Il offre la possibilité d'indexer et de requêter des données.
- Des extensions préconfigurées de l'interface utilisateur Web sont disponibles. Il est possible de les adapter aux cas d'utilisation en les modifiant.
- Pour le contrôle d'intégrité des fichiers, Wazuh utilise SHA256 au lieu de SHA1 et MD5.

Détection d'intrusion

- La taille maximale des logs passe de 6KB (Ossec) à 64KB (Wazuh).
- Ajout de nouvelles règles d'analyse des logs mais aussi de nouveaux décodeurs. Ces règles sont mises à jour régulièrement.
- Présence de règles pour Suricata, utilisant le décodeur JSON pour le fichier « eve.json ».

Modules

- Création d'un module permettant de faire l'inventaire software/hardware pour chaque agent.
- Création d'un module d'intégration VirusTotal. La combinaison de cet outil avec le moteur de contrôle d'intégrité des fichiers fournit un moyen simple de scanner les fichiers spécifiés afin de les inspecter à la recherche de contenus malicieux.
- Création d'un module Osquery pour des recherches sur des bases de données.

Détection de vulnérabilités

- Création d'une base de données de CVE mis à jour avec le repository OVAL. Le couplage du module d'inventaire à cette base de données permet de détecter des logiciels devenus vulnérables.
- Création d'un module OpenSCAP. Les politiques OpenSCAP définissent les exigences auxquelles tous les systèmes d'une organisation doivent satisfaire afin d'être en conformité avec les politiques de sécurité.

Le service Suricata

Introduction

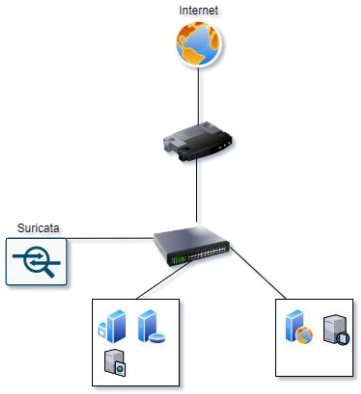
Suricata est un NIDS créé en 2008 par Victor Julien qui appartient désormais à la fondation OISF (Open Information Security Foundation).

L'une des tâches les plus importantes de Suricata est la reconstruction du trafic. Suricata fonctionne extrêmement bien avec l'inspection approfondie des paquets et la comparaison de modèles, ce qui le

rend incroyablement utile pour la détection des menaces et des attaques. Chacune des couches du modèle OSI doit être reconstruite correctement et cela induit la gestion de la fragmentation IP, de la segmentation TCP ainsi que de l'ensemble des types d'invalidités (incohérence dans les longueurs annoncées, somme de contrôle non correcte, taille de fenêtre TCP invalide). La grande force de Suricata est la détection automatique de protocole (TCP/UDP, ICMP, DNS, HTTP …) même quand ceux-ci ne communiquent pas à travers les ports standards.

Architecture

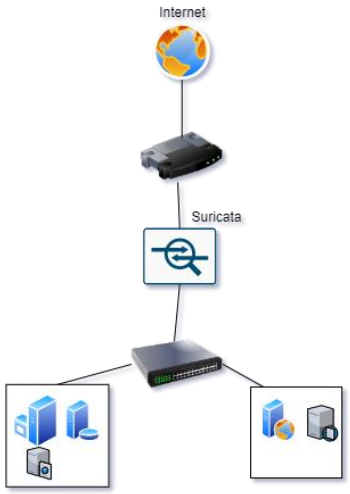
Suricata peut être utilisé comme NIDS. Par exemple, il est possible de placer la sonde réseau dans une VLAN d'un switch qui permet d'accéder à une VLAN à risque, de copier tous les paquets qui arrivent sur ce switch à la sonde réseau et de remonter les alertes.



Il est aussi possible d'utiliser Suricata comme NIPS. Les règles Suricata peuvent réaliser 4 actions :

- pass : Suricata arrête de scanner le paquet courant.
- drop : Suricata stoppe le paquet immédiatement + création une alerte.
- reject : Suricata créé une alerte. Dans le cas d'une connexion TCP, il envoie un paquet RST à l'émetteur et au receveur du paquet initial.
- alert : Suricata créé une alerte.

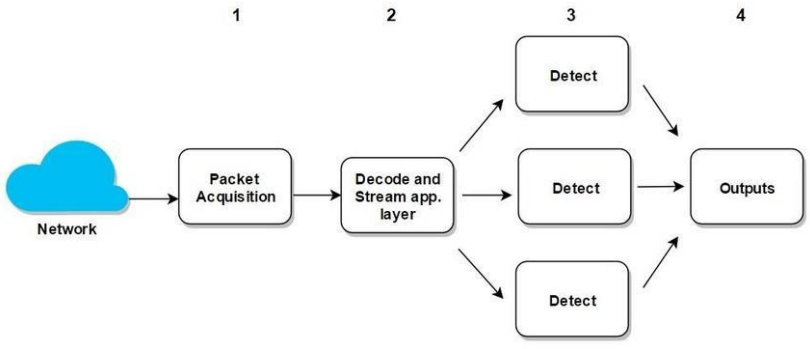
On peut faire transiter tous les paquets reçus à travers la sonde après analyse de ces derniers.



Performance

Le trafic réseau augmentant considérablement, la nature multi-threads de Suricata permet un scaling très facile en ajoutant des threads de traitement de paquets lorsque le volume de trafic le rend nécessaire. Suricata possède 4 threads distincts :

- Thread d'acquisition des paquets, il est responsable de la lecture des paquets sur le réseau.
- Thread de décodage, il décode les paquets.
- Thread de détection, il compare les signatures des paquets. C'est cette partie qui peut être exécutée dans plusieurs threads.
- Thread de sorties, il est responsable du traitement des alertes



.

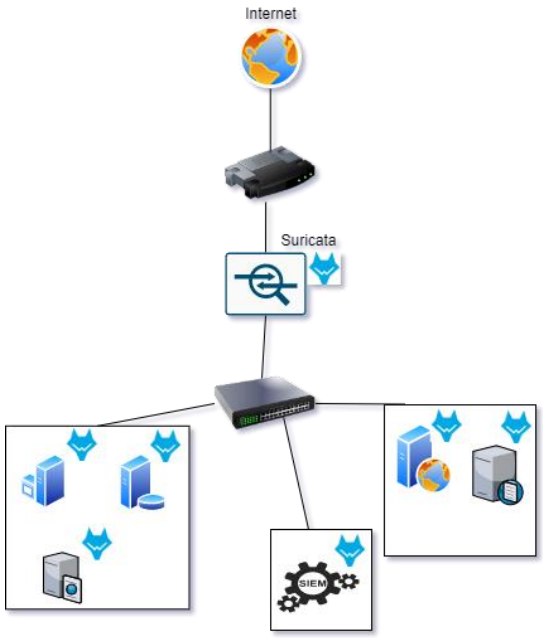
L'intérêt de Suricata est en parti dû à cette nature multi-threads qui augmente sa performance considérablement.

Combinaison NIDS / HIDS

Imaginons une architecture avec un SIEM (mis en place avec la suite ELK, par exemple), un plug-in d'un HIDS (Wazuh par exemple) va jouer le rôle de serveur sur le SIEM. Des agents sont dispersés sur des serveurs à risque (Web, AD, FTP, DNS …) mais aussi sur les NIDS (sondes réseau Suricata par exemple). La couplage entre NIDS et HIDS peut être intéressant à plusieurs niveaux.

Réseau et compatibilité

La normalisation des logs envoyés à travers le réseau peut être un plus avec le couplage NIDS / HIDS. Seul un HIDS peut envoyer des alertes au SIEM, ce qui uniformise les envois de logs à travers le réseau. Il sera donc possible de gérer tous les logs reçus avec un seul module. Le service Wazuh peut analyser n'importe quel type de logs à l'aide d'un décodeur approprié.



Réponse active brutale du NIPS

Les faux-positifs sont monnaie courante pour un NIDS/NIPS. La disponibilité étant un point essentiel dans la mise en place de sécurité sur un réseau, il faut faire très attention lorsqu'il y a une réponse active directement sur les paquets qui traverse le réseau. Mieux vaut une response active d'un HIPS pour donner suite à une alerte d'un NIPS plutôt que des actions directement sur les paquets.

Conclusion

Les améliorations de Wazuh par rapport à Ossec sont considérables. Wazuh est un outil extrêmement puissant, très adaptable à son environnement et qui s'améliore de jour en jour. De son côté, Suricata offre, par sa compréhension des protocoles et ses fonctionnalités d'extraction, des possibilités intéressantes. Ces services sont open-source, gratuits et possèdent une énorme communauté, ce qui est très important pour sa durabilité et sa transparence. Le couplage de ces technologies peut être très avantageux, tant pour la pérennité du réseau que pour l'administrateur qui se charge de cette dernière.