MASTER 2 INFORMATIQUE – CYBERSÉCURITÉ

Projet SIEM – Groupe QRadar / Collecte de logs

Université Paris Cité

Compte rendu sur la partie QRadar

Ivan KRIVOKUCA (22306432)

M. Rodrigue PARAISO

14 avril 2025

Table des matières

Introduction et contextualisation du projet

Dans le cadre de la matière Projet SIEM, notre équipe est responsable de l'implémentation d'IBM QRadar comme solution SIEM et de l'infrastructure de collecte centralisée des journaux d'événements. Cette mission s'intègre dans l'architecture globale décrite dans le Document d'Architecture Technique (DAT).

Ce rapport se concentre exclusivement sur notre déploiement de QRadar, qui a subi deux phases distinctes :

1. **Phase initiale** : Déploiement de la version 7.3.3 sur nos machines personnelles, adapté aux limitations matérielles individuelles, notamment en mémoire vive. Cette étape préliminaire nous a permis de maîtriser les fondamentaux de QRadar.
2. **Phase de production** : Migration vers la version 7.5 depuis le 4 avril 2025, rendue possible grâce à la VM fournie par l'équipe d'architectes (4 cœurs CPU, 30 Go RAM, 300 Go stockage).

Le choix de la version 7.5 présente deux avantages techniques majeurs :

- Intégration des modules de visualisation QRadar Pulse et QRadar Use Cases, répondant aux exigences notamment niveau dashboarding.
- Device Support Modules (DSM) récemment actualisés, éliminant les procédures manuelles de mise à jour nécessaires avec la version 7.3.3.
- Version à jour et beaucoup plus stable, retour d’expérience après l’avoir manipulé pendant cette phase de production.

Les DSM dans notre architecture QRadar

Les Device Support Modules (DSM) constituent un élément central de notre déploiement QRadar, permettant la normalisation des logs provenant des diverses technologies de notre infrastructure. Cette normalisation transforme les données brutes en format exploitable, condition préalable essentielle aux fonctions de corrélation et d'analyse de notre SIEM.

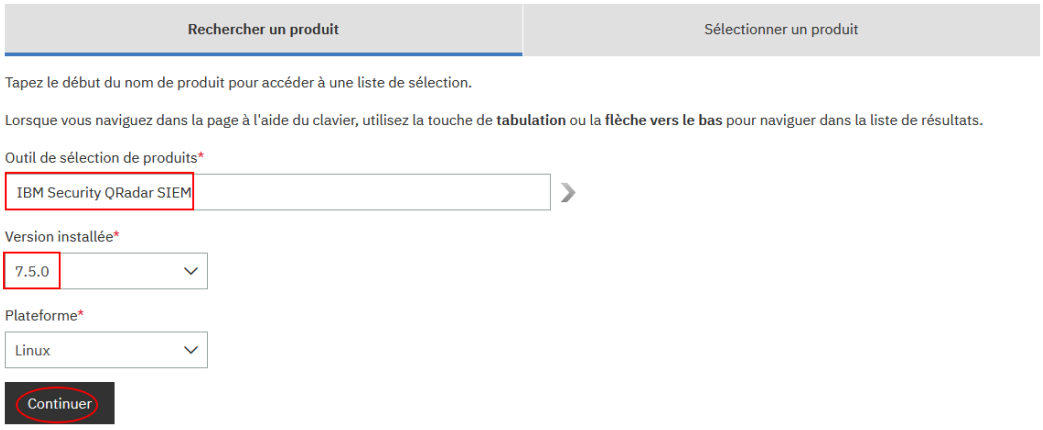
Processus d'installation des DSM

L'ensemble des DSM utilisés ont été téléchargé (ou mise à jour) depuis la plateforme IBM Fix Central : https://www.ibm.com/support/fixcentral/

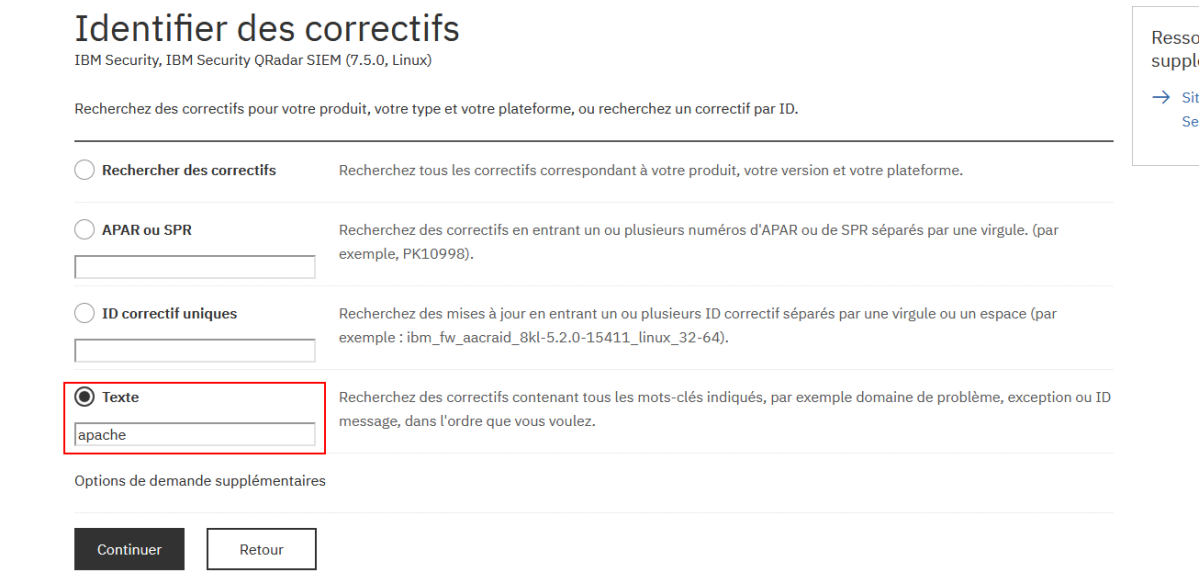
Vérification des versions actuelles (sur la machine QRadar) :

```
rpm -qa | grep -i nameofDSM
```

Sur la plateforme IBM Fix Central

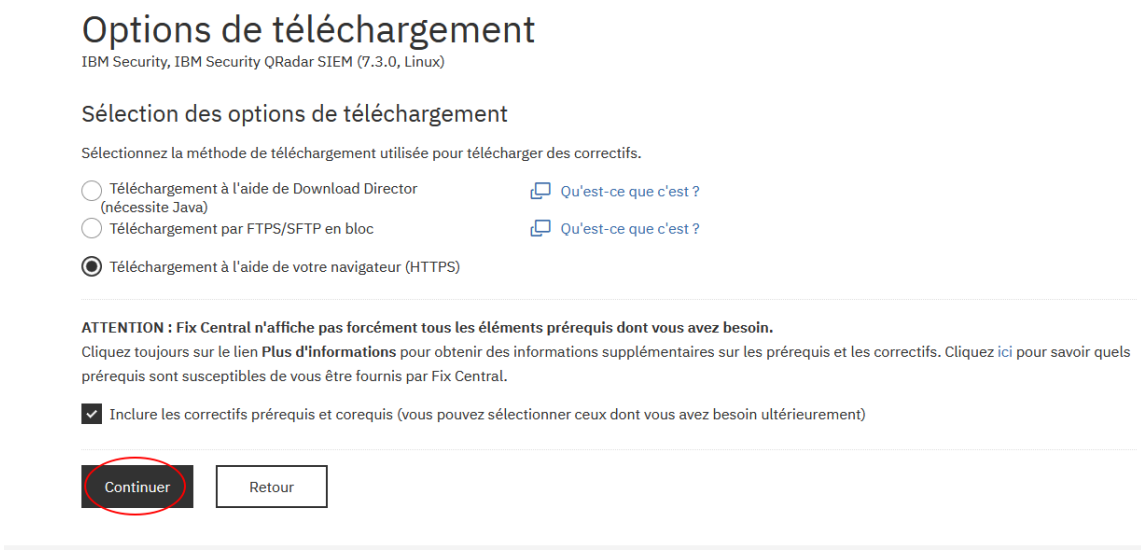


Ici, nous avons fait la recherche pour récupérer le dernier DSM pour Apache



Ensuite, pour l'installation de chaque DSM, nous avons suivi le processus d'installation suivant :

1. Téléchargement du fichier *rpm*



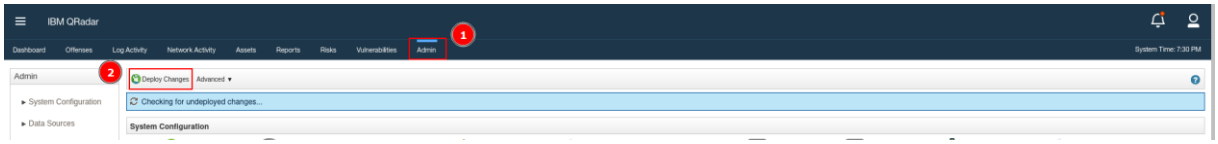
2. Transfert vers le serveur QRadar :

```
scp [NOM_DSM] root@[IP_QRADAR]:~/
```

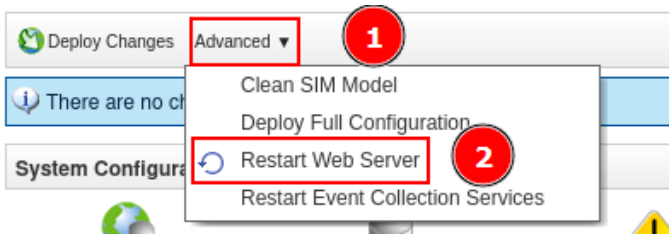
3. Puis sur la machine QRadar :

```
yum -y install [NOM_DSM]
```

4. Déploiement des changements via l'interface d'administration (option "Deploy Changes" dans la section "Admin")



Redémarrage des services web pour finaliser l'installation



La liste des DSM qui ont été téléchargés sont :

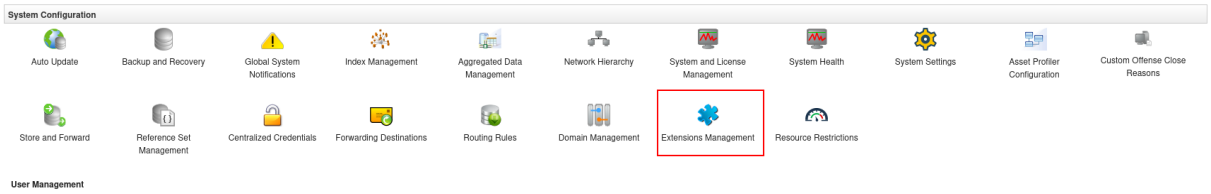
- 7.5.0-QRADAR-DSM-Suricata-7.5-20230215062721.noarch.rpm
- 7.5.0-QRADAR-DSM-NetgatePfSense-7.5-20240918094805.noarch.rpm
- 7.5.0-QRADAR-DSM-ApacheWebserver-7.5-20230727093436.noarch.rpm

Configuration via plateforme IBM X-Force Exchange

La plateforme IBM X-Force Exchange nous a permis d'enrichir notre déploiement avec les modules complémentaires suivants :

- QRadar Pulse, pour les dashboards
- QRadar Use Case Manager
- Des customs properties pour les DSM :
    - Apache
    - Snort
    - ~~Squid~~ (pas nécessaire vu que le format de log n'est pas celui standard)

Après les avoir téléchargés, l'installation de ses modules se fait dans « Extensions Managment » qui est présent dans l'interface administrateur.



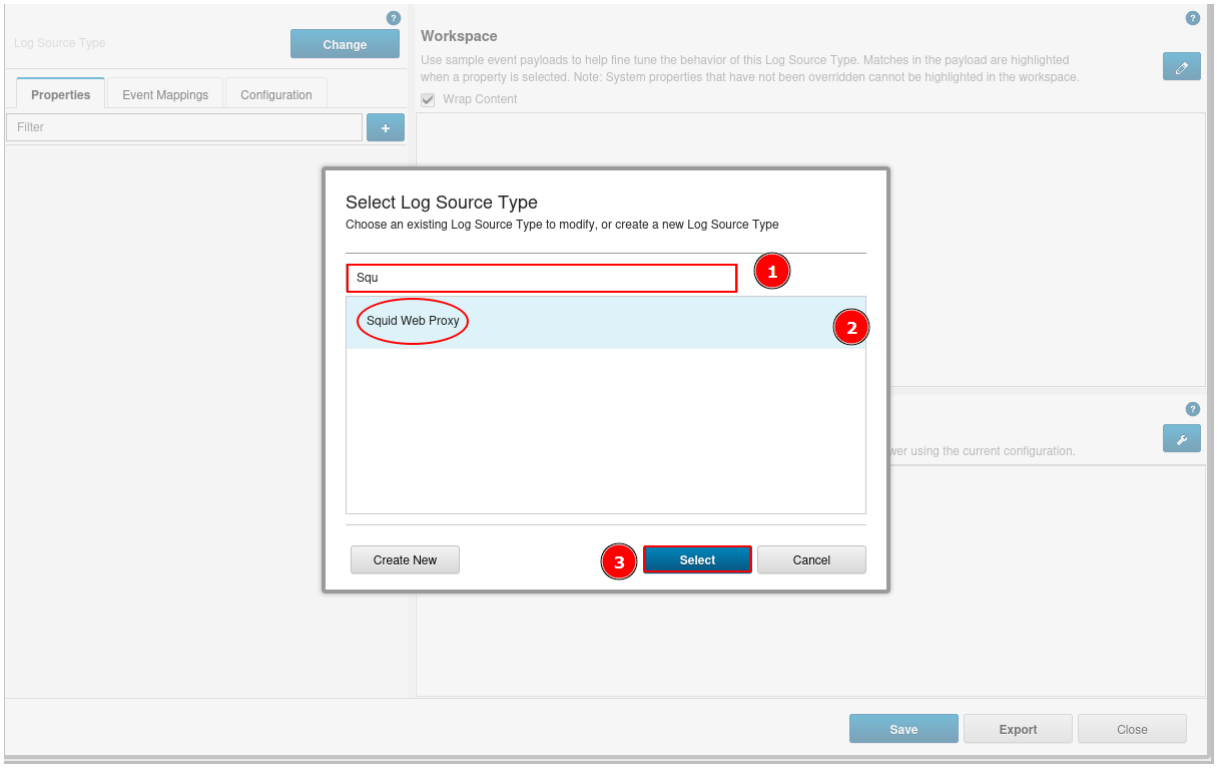
Développement de parseurs personnalisé et des Events Mappings

Nos configurations de parsing ont été réalisées via l'outil DSM Editor accessible dans l'interface d'administration de QRadar. Cette étape est critique car au-delà du simple parsing, la catégorisation des logs détermine leur exploitation dans les analyses et tableaux de bord.

Un log correctement parsé mais non catégorisé apparaît comme "Unknown log event", limitant considérablement son utilité dans la corrélation et la détection d'incidents.

Modifications des DSM

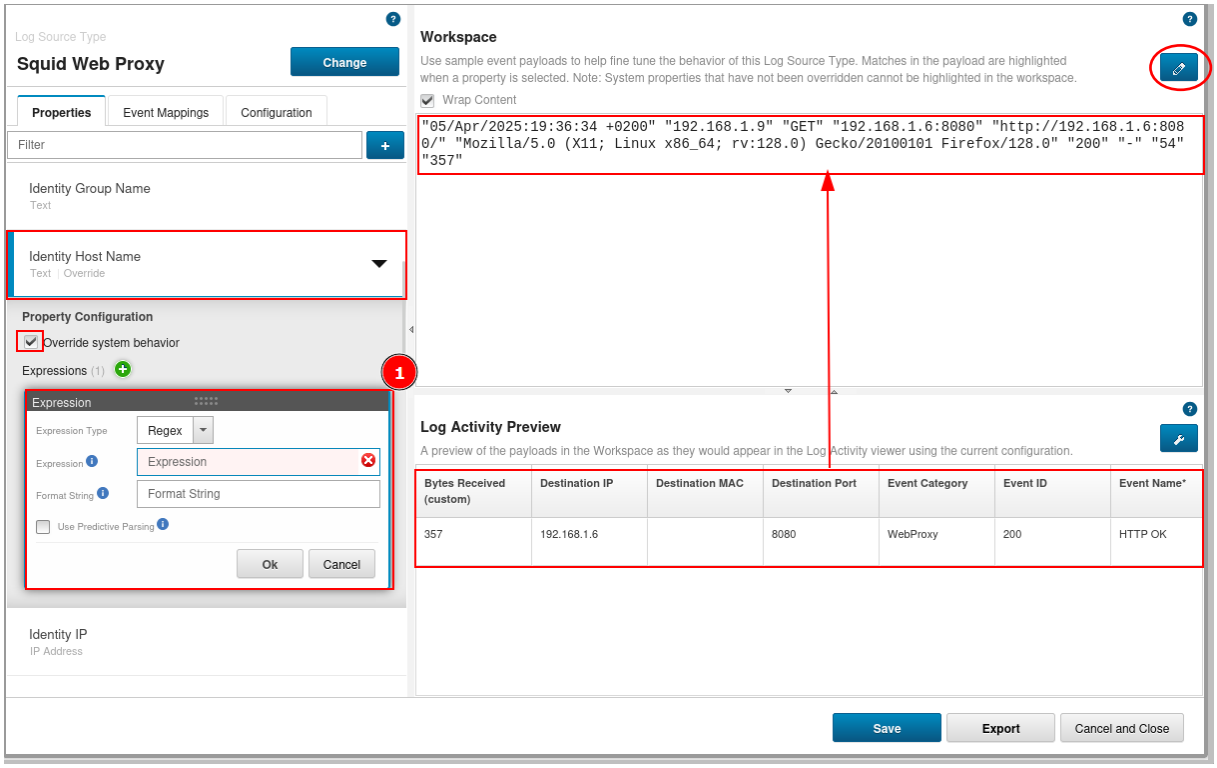
1. Dans l'interface d'administration QRadar, accédez à l'outil DSM Editor
2. Dans la barre de recherche, saisissez le nom du DSM à modifier et sélectionnez le DSM



Personnalisation des expressions régulières

Pour modifier une extraction de champ :

1. Sélectionnez la propriété à modifier dans la liste des champs disponibles
2. Cliquez sur Override system behavior pour remplacer le comportement par défaut
3. Définissez l'expression régulière adaptée au format de log spécifique
4. Configurez le Format String approprié pour l'extraction des données

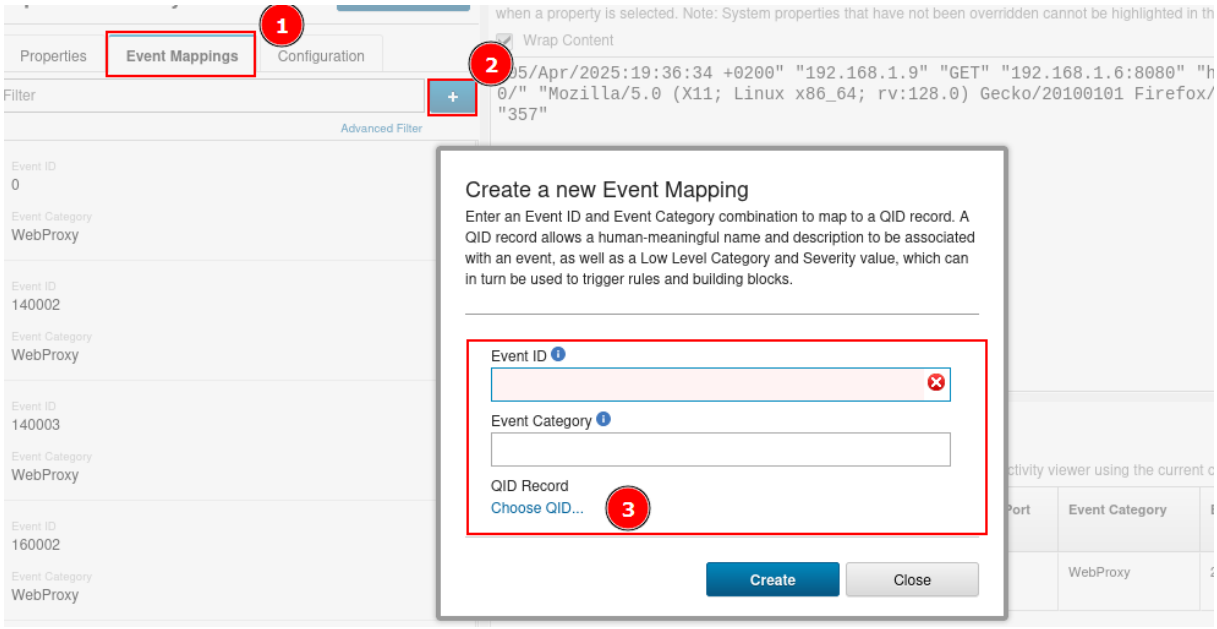


Configuration des mappings d'événements (Event Mapping)

1. Identifiez l'Event ID et l'Event Category provenant du log source
2. Dans l'onglet Event Mapping, associez ces valeurs à un QID existant
3. Si le QID n'existe pas, créez-en un nouveau en spécifiant :

- Le nom descriptif de l'événement
- Sa description détaillée
- La catégorie haute et basse correspondante

- Le niveau de sévérité approprié



Netgate pfSense et Suricata

Le DSM installé pour pfSense est fonctionnel, assurant à la fois :

- Un parsing correct des logs
- Un mapping fonctionnel des ID d'événements

Celui-ci a été confirmé après avoir reçu un log venant de cet équipement.

Suricata

L'intégration de Suricata a nécessité une approche différente en raison de l'absence de retour de l'équipe responsable (que ça soit pour un échantillon de log ou les règles).

Nous avons procédé tout de fois de la manière suivante :

- Base de référence : Utilisation d'un échantillon de message par IBM (https://www.ibm.com/docs/fr/dsm?topic=suricata-sample-event-message)

- Personnalisation du DSM avec les expressions régulières suivantes :

Event Category	"category":"(.*?)"
Event ID	"signature_id":"(\d+)"
URL HTTP	"url":"([^"]+)"
User Agent HTTP	"http_user_agent":"([^"]+)"
Status HTTP	"status":(\d+)

Le parsing fonctionne correctement, mais le mapping des événements reste incomplet faute de connaissance des règles spécifiques implémentées.

Le DSM Suricata ne fournit pas par défaut de mapping entre ID et types d'événements, sans connaissance des règles spécifiques implémentées, nous n'avons pas pu finaliser cette étape.

Squid

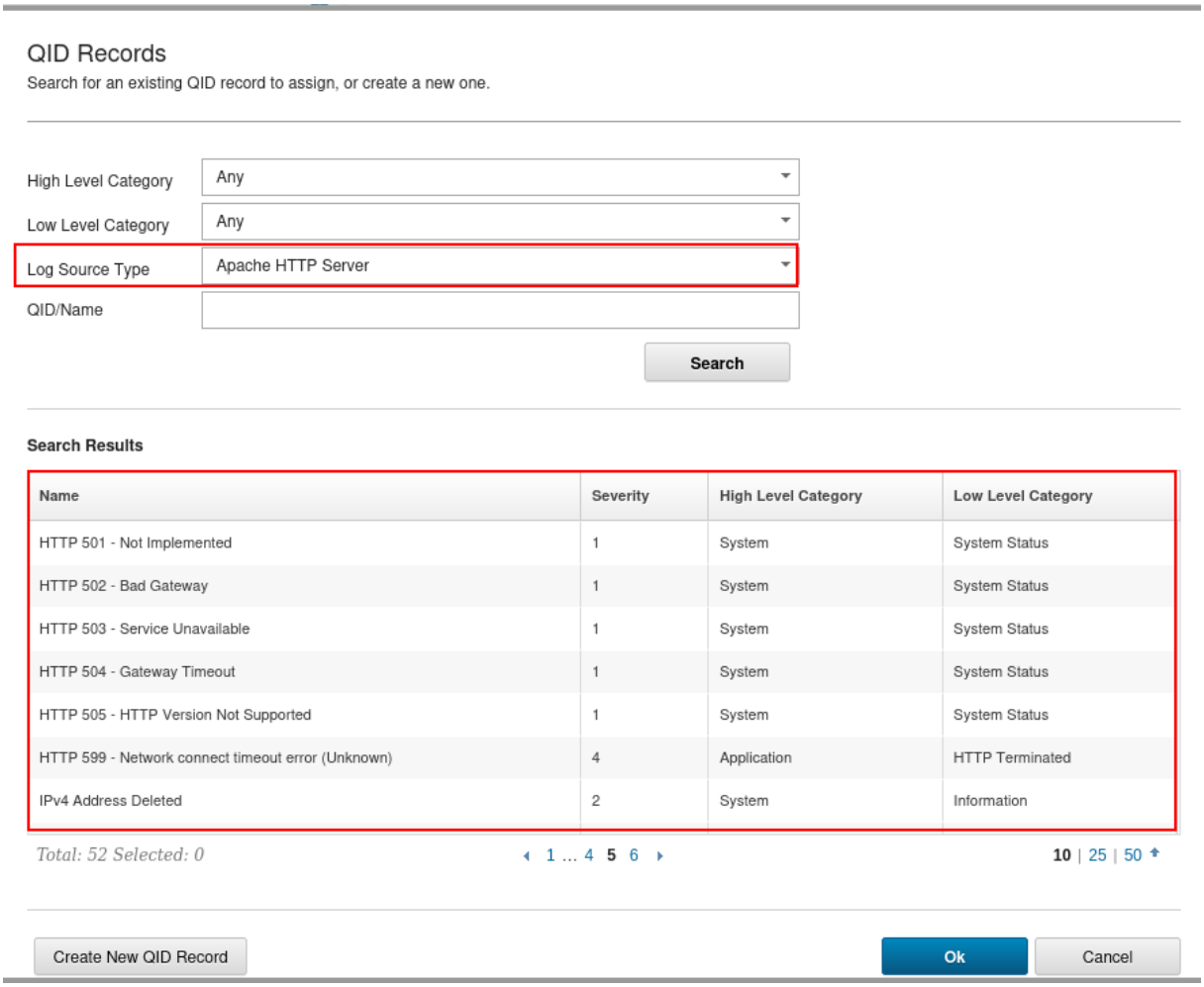
Nous avons reçu le format de log suivant venant de l'équipe Squid :

```
"05/Apr/2025:19:36:34 +0200" "192.168.1.9" "GET" "192.168.1.6:8080"
"http://192.168.1.6:8080/" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0)
Gecko/20100101 Firefox/128.0" "200" "-" "54" "357"
```

Le format reçu de l'équipe Squid différant significativement du format standard, nous avons dû adapter le DSM avec de nouvelles expressions régulières, dont :

Destination IP	"GET"\s+"([0-9.]+):
Source IP	^"[^"]+"\s+"([^"]+)"
Destination Port	"GET"\s+"[0-9.]+:([0-9]+)"
Protocole	[^"]*"[^"]*"[^"]*"[^"]*"[^"]*"([^:]+)://
Bytes Received	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([0-9]+)"
Method	^"[^"]+"\s+"[^"]+"\s+"([^"]+)"
Response Code	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([0-9]+)"
Response Time	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([^"]+)"
URL	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([^"]+)"
URL Host	http://([^:/]+)
URL Path	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"https?://[^/]+(/[^"?]*)"
URL Query String	[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+\?([^"]+)"
URL Scheme	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([^:]+)://
ID Event	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"(\d+)"
Log Source Time	"([^"]+)" format date : dd/MMM/yyyy:HH:mm:ss Z
Event ID	"(2\d{2}|3\d{2}|4\d{2}|5\d{2})"
Useragent	squid\[\d+\]:.*?".*?".*?".*?".*?".*?"([^"]+)"
Cookie HTTP	"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"[^"]+"\s+"([^"]+)"

Pour le mapping d'événements, nous avons réutilisé la structure des événements HTTP d'Apache, adaptée aux codes de réponse Squid.



Snort

Après réception d'échantillons de logs de l'équipe Snort, nous avons confirmé que le DSM standard fonctionnait correctement sans modification supplémentaire.

EDR / Wazuh

En l'absence de DSM préexistant pour Wazuh, nous avons créé un parseur complet basé sur les logs fournis :


```
{"timestamp":"2025-04
02T18:27:10.401+0000","rule":{"level":3,"description":"Wazuh server
started.","id":"502","firedtimes":1,"mail":false,"groups":["ossec"],
"pci_dss":["10.6.1"],"gpg13":["10.1"],"gdpr":["IV_35.7.d"],"hipaa":[
"164.312.b"],"nist_800_53":["AU.6"],"tsc":["CC7.2","CC7.3"]},"agent"
:{"id":"000","name":"4320de26bcd1"},"manager":{"name":"4320de26bcd1"
},"id":"1743618430.0","full_log":"ossec: Manager
started.","decoder":{"name":"ossec"},"location":"wazuh-monitord"}
```


Voici les catégories qui ont été implémentés :

Event ID	"rule":\s*\{\s*"level":\s*\d+,\s*"description":\s*"[^"]*",\s*"id":\s*"(\d+)"
Event Category	"groups":\s*\[\s*"([^"]*)"
Destination IP	"agent":\s*\{[^}]*"ip":\s*"([^"]*)"
Identity Host Name	"agent":\s*\{[^}]*"name":\s*"([^"]*)"
Identity Group Name	"groups":\s*\[(.*?)\]
Identity Extended Field	"description":\s*"([^"]*)"
Identity IP	"manager":\s*\{[^}]*"name":\s*"([^"]*)"
Log Source Time	"timestamp":\s*"([^"]*)" --> yyyy-MM-dd'T'HH:mm:ss.SSSZ
Source IP	"agent":\s*\{[^}]*"ip":\s*"([^"]*)"
Source Port	"data":\s*\{[^}]*"port":\s*(\d+)
Protocol	"decoder":\s*\{[^}]*"name":\s*"([^"]*)"
Username	"data":\s*\{[^}]*"srcuser":\s*"([^"]*)"
Wazuh Command	"data":\s*\{[^}]*"command":\s*"([^"]*)"
Wazuh Destination User	"data":\s*\{[^}]*"dstuser":\s*"([^"]*)"
Wazuh Full Log	"full_log":\s*"(.*?)"
Wazuh Location	"location":\s*"([^"]*)"
Wazuh Rule Level	"rule":\s*\{[^}]*"level":\s*(\d+)

Pour les Event mapping, nous nous sommes basés sur ceux qui ont été envoyés, si on reçoit des ID inconnus, QRadar va les stocker pour qu'on puisse les mapper plus tard.

Event ID	Event Category	Name	Description	Log Source Type	High Level Category	Low Level Category	Severity
502	ossec	Wazuh Server Started	Détection du démarrage du service serveur Wazuh	Wazuh	System	Service Start	3
503	ossec	Wazuh Agent Started	Détection du démarrage d'un agent Wazuh sur un système distant	Wazuh	System	Service Start	3
510	ossec	Host-based Anomaly Detection	Détection rootcheck (p-e compromission système)	Wazuh	Suspicious Activity	Suspicious File Name	7
5501	pam	PAM Login Session Opened	Ouverture de session PAM pour un utilisateur du système	Wazuh	Authentication	Login Success / User Login Success	3
5502	pam	PAM Login Session Closed	Fermeture d'une session utilisateur PAM sur le système	Wazuh	Authentication	Logout / Host Logout	3
5403	syslog	First Time User Executed Sudo	Première exécution de sudo par un utilisateur, potentiellement suspect	Wazuh	Suspicious Activity	Suspicious Activity	4
5402	syslog	Successful Sudo to ROOT	Élévation de privilèges vers ROOT via sudo	Wazuh	Authentication	Privilege Escalation	3
31101	web	Web Server 400 Error Code	Détection d'erreur 400 sur serveur web, possible tentative d'attaque	Wazuh	Potential Exploit	Web Attack	5

Configuration des Log Sources

QRadar nécessite une identification précise des sources de logs pour optimiser le traitement et l'analyse des événements. Deux paramètres sont essentiels pour cette identification :

- L'adresse IP de l'équipement émetteur
- Le type d'équipement concerné (DSM correspondant)

Bien que QRadar puisse parfois déterminer automatiquement le type d'équipement grâce au format des logs reçus, une configuration explicite garantit un traitement optimal et évite le stockage d'événements non catégorisés.

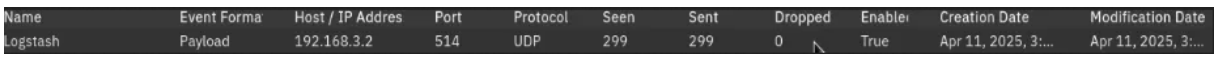
Dans notre architecture, Logstash/Kafka (192.168.3.2) est utilisé comme point central de collecte et de redistribution des logs. Cette approche présente un avantage architectural significatif : QRadar ne reçoit des logs que d'une source unique. Nous avons déclaré dans QRadar les différents types d'équipements dont les logs transitent par Logstash, permettant leur identification correcte dans la section "Log Sources" de l'interface d'administration.

Envoi des logs QRadar vers Kafka

Nous avons configuré QRadar pour envoyer ses propres logs vers Kafka via l'outil "Event Forwarding". Trois catégories de logs sont transmises :

1. **SIM Audit-2 :: qradar** - Contient les logs d'audit essentiels (connexions utilisateurs, modifications de configuration, actions administratives)
2. **System Notification-2 :: qradar** – Notifications systèmes (démarrage/arrêt des services, alertes système)
3. **Health Metrics-2 :: qradar** - Fournit des informations sur l'état de santé du système QRadar

Preuve que QRadar a envoyé certains de ses logs :



Name	Event Forma	Host / IP Addres	Port	Protocol	Seen	Sent	Dropped	Enable	Creation Date	Modification Date
Logstash	Payload	192.168.3.2	514	UDP	299	299	0	True	Apr 11, 2025, 3:...	Apr 11, 2025, 3:...

Gestion du cycle de vie des logs et politique d'accès

Conformément aux exigences définies dans le DEX, nous avons paramétré la rétention des logs à 30 jours, valeur par défaut (contre les 60 jours initialement prévus). Ce choix optimise l'utilisation de l'espace disque tout en maintenant une période d'historique suffisante pour les analyses.

Pour la politique d'accès, outre le compte "admin" disposant de privilèges complets, nous avons créé un profil utilisateur restreint "User" avec les droits suivants :

- Lecture seule sur les logs entrants
- Accès en consultation aux tableaux de bord

Dashboards

Pour les Dashboards, nous avons utilisé l'outil QRadar Pulse, en ajoutant un nouveau dashboards qu'on a appelé : *Vue d'ensemble SOC*

Composition du tableau de bord principal

- **Active offenses over time** : Visualisation temporelle de l'évolution des incidents actifs
- **Most recent offenses** : Affichage des dernières alertes détectées
- **Most severe offenses** : Identification rapide des incidents les plus critiques
- **Number of critical/high offenses** : Compteurs des incidents par niveau de sévérité

Analyse des sources et catégories

- **Top offense categories** : Distribution des incidents par catégorie
- **Top 10 log sources** : Sources générant le plus d'événements
- **Number of events per user** : Répartition des événements par utilisateur

En complément du tableau de bord personnalisé, nous avons déployé QRadar Use Case Manager et installer des cas d'usage spécifiques à chaque type d'équipement intégré dans notre architecture (PFsense, Suricata et Snort).