ALGORITHME SOLUTIONS INC.

Analyse de sécurité avec la journalisation

Rédigé par :
M. Donavan Martin
Candidat à la profession d'ingénieur (oiq.qc.ca)

Version non-finale (public)
Sherbrooke – 22 juin 2021

Table des matières

1 Introduction

Connaître les techniques offensives est le meilleur moyen de se prémunir des attaques informatiques. Dans le domaine de la cybersécurité, une excellente journalisation permet de récolter beaucoup d'informations sur les techniques utilisées lors des tentatives. L'analyse des tentatives d'intrusion, d'injection et des récoltes d'informations est primordiale pour défendre un système informatique. Dans cette publication, nous abordons des concepts d'analyse des journaux avec des exemples concrets à titre de sensibilisation envers la cybersécurité.

2 La journalisation

Tout dépendamment de la configuration du serveur, les journaux peuvent se situer à plusieurs endroits. En effet, la configuration des processus permet généralement de déplacer le fichier de journalisation dans un répertoire différent de celui par défaut. Dans les exemples qui suivent, la configuration par défaut d'un serveur Ubuntu est utilisée.

2.1 Composition d'une entrée journal

Conventionnellement, l'entrée dans un journal comprend une date, une heure, un utilisateur, un processus, un état et la description sur l'événement.

```
<Date><Temps><Utilisateur><Processus><Etat><Evenement>
```

2.2 Exemple d'analyse de journaux de processus

La liste des journaux et répertoires de journaux des processus par défaut peut être affiché à l'aide de la commande suivante :

```
ls -lt /var/log/
```

Prenons un exemple pour analyser les 10 dernières entrées du syslog.

```
tail -10 /var/log/syslog
Jun  9 00:23:03  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=92.118.161.1 DST=143.110.xxx.xxx <...>
Jun  9 00:23:43  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=45.134.26.50 DST=143.110.xxx.xxx <...>
Jun  9 00:24:14  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=159.89.150.163 DST=143.110.xxx.xxx <...>
Jun  9 00:24:37  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=45.134.26.56 DST=143.110.xxx.xxx <...>
Jun  9 00:24:38  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=103.145.13.26 DST=143.110.xxx.xxx <...>
Jun  9 00:24:38  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=89.248.165.104 DST=143.110.xxx.xxx <...>
Jun  9 00:24:56  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=181.214.206.101 DST=143.110.xxx.xxx <...>
Jun  9 00:25:16  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=45.143.203.16 DST=143.110.xxx.xxx <...>
Jun  9 00:25:37  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=45.134.26.55 DST=143.110.xxx.xxx <...>
Jun  9 00:26:03  <...> [UFW BLOCK] IN=eth0 OUT= MAC=b2:a5:8a:27:65:6e:fe:<...> SRC=162.218.65.219 DST=143.110.xxx.xxx <...>
```

On peut rapidement observer qu'il y a un problème avec le bannissement. En fait, le pare-feu ne bannit pas la MAC adresse et aucun autre processus banni ces tentatives d'intrusion. À noté que Fail2Ban est utilisé, mais le bannissement par adresse MAC n'est pas une fonctionnalité de Fail2Ban. Il faut donc trouver la source du problème. Puisque ces logs ressemblent à une tentative par force brute, nous pouvons analyser le fichier d'authentification :

```
tail -10 /var/log/auth.log
Jun  9 10:53:51 <...> sshd[97108]: Failed password for invalid user vagrant from 209.141.37.3 port 35568 ssh2
Jun  9 10:53:52 <...> sshd[97107]: Failed password for invalid user ubuntu from 209.141.37.3 port 35562 ssh2
Jun  9 10:53:52 <...> sshd[97104]: Failed password for root from 209.141.37.3 port 35560 ssh2
Jun  9 10:53:52 <...> sshd[97106]: Failed password for invalid user test from 209.141.37.3 port 35564 ssh2
Jun  9 10:53:52 <...> sshd[97105]: Failed password for invalid user user from 209.141.37.3 port 35566 ssh2
Jun  9 10:54:46 <...> wordpress(<domain>)[96225]: Authentication failure for user1 from 172.70.98.50
Jun  9 10:55:00 <...> wordpress(<domain>)[96226]: Authentication failure for user2 from 172.69.34.238
```

Le log d'authentification montre deux processus qui génère des entrées journaux avec des tentatives d'intrusion soit sshd et wordpress. Enfin, à partir de ces informations nous pouvons augmenter la sécurité en configurant les deux processus.

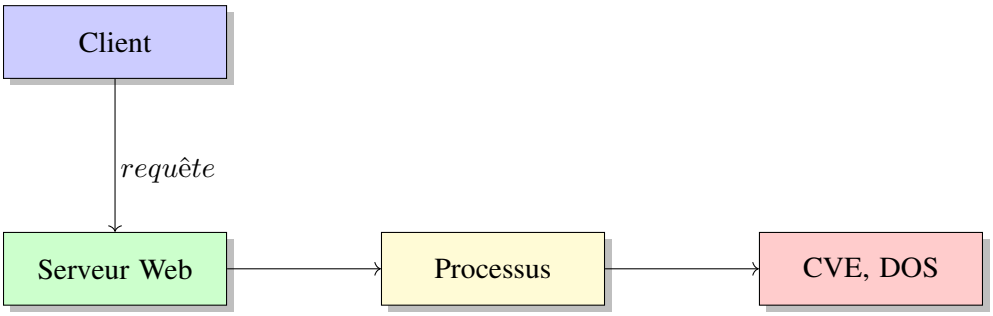

FIGURE 2-1 – Exemple de schéma d'analyse des journaux de processus

2.3 Exemple de journaux d'analyse de requête

Les deux types de requêtes les plus utilisés sont le type GET et le type POST. Le type GET prend des paramètres (appelé params) dans l'URL. Par exemple, la requête GET *www.google.ca?cookie=123* comprend le paramètre cookie dont la valeur est 123. Sans si limité, la réponse d'une requête GET peut être une page html, une chaine de caractères ou un object JSON. En règle générale, la requête GET sert a récupérer de l'information dans un l'état des *params*.

Par convention, la requête POST sert généralement à ajouter, modifier et supprimer des informations sur le serveur. Pour ce faire, la requête POST utiliser généralement le *body* de la requête pour envoyé de l'information au serveur.

L'un des problèmes qui peut subvenir avec ces deux types de requêtes est l'injection. Si les *params* ou le *body* ne sont pas analysés correctement, alors l'injection est probable. Une solution simple pour valider *params* ou le *body* est d'ajouter un valideur de schéma de données. Prenons un exemple avec un code d'une requête AJAX sur un serveur Wordpress :

```
add_action( 'wp_ajax_endpoint', 'save_secret_info');
function save_secret_info() {
  if ( !is_admin()
```

```
    || !isset($_POST['secret'])
    || !wp_verify_nonce($_POST['secret'], 'algosol-nonce')) {
   $output = "Permission denied: nonce invalid";
    log_ajax($output );
   wp_die("permission denied");
  }else {
   if(is_string($_POST['secret'])){
     // A more secure way to save secret
     log_ajax("saved" );
     wp_die("saved");
   }else{
     log_ajax("secret n'est pas une chaine de caracteres");
     wp_die("invalid format");
   }
  }
}

function log_ajax($message){
 $date = date('Y-m-d H:i:s');
 $user = get_current_user ( );
 $process = "AJAX";
 error_log( $date. " ". $user. " ". $process." ". $message);
}
```

Selon le code précédent, chaque condition de validation de la requête est enregistrée dans les journaux. Cet exemple démontre la base d'une validation par schéma de données, mais cet exemple pourrait être beaucoup plus précis.

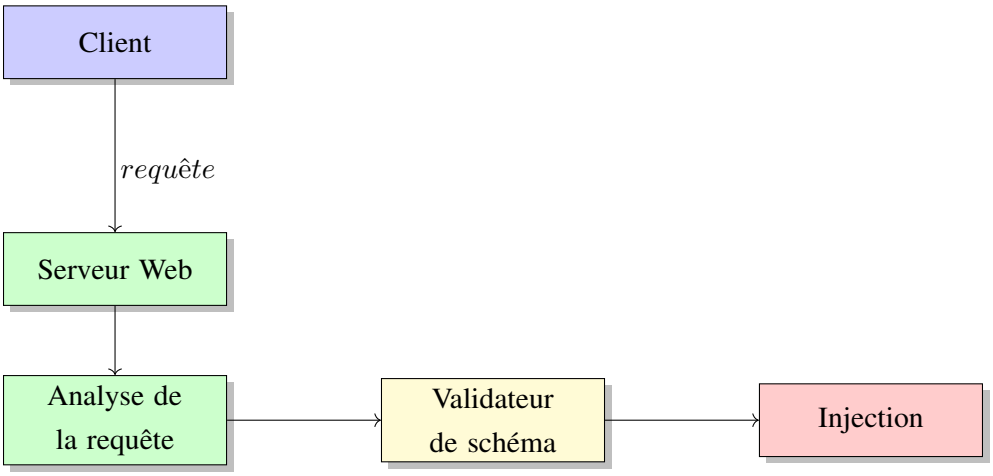

FIGURE 2-2 – Exemple de schéma d'analyse des journaux de requêtes

2.4 Exemple de journaux d'analyse de trace de pile

lorsqu'une requête malicieuse est envoyée au serveur, de nombreuses conditions sont à prévoir. Il peut arriver que certaines conditions ne soient pas validées par les processus et l'étape de l'analyse de la requête. Dans ce cas, l'exécution de la requête peut mener à des comportements indésirables du programme. Si le code source du programme n'est pas connu de l'attaquant, alors les tentatives auront la forme de probabilité. Par exemple, la réussite d'un Shellcode envoyé dans une requête est plus difficile en raison des contraintes d'exécution et il est donc plus probable que plusieurs tentatives seront nécessaires.

Lorsqu'une requête mal analysée est exécutée par un système informatique, l'exécution de la requête génère fréquemment des erreurs d'exécution. Les erreurs d'exécution sont généralement affichées dans la trace de pile (Stacktrace) et les langages de programmation permettent aussi de générer la trace de pile ou de générer une exception.

Une fois de plus, les journaux peuvent aider à détecter ce type d'événement. D'ailleurs, la trace de pile comprend souvent suffisamment d'information pour corriger le problème par un programmeur.

Voici un exemple pour afficher la trace de pile :

```
var_dump(debug_backtrace());
```

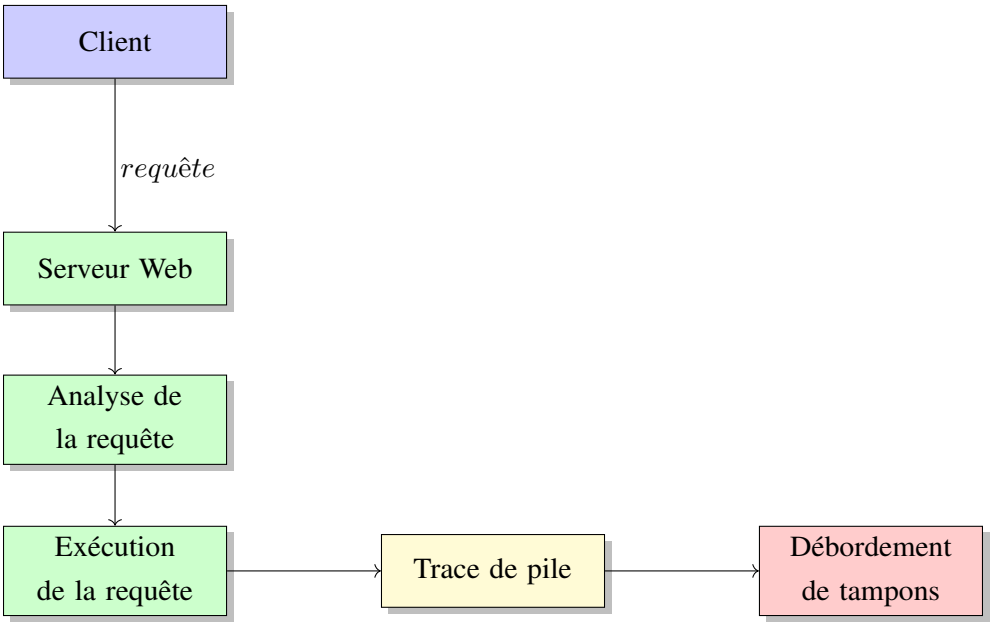

FIGURE 2-3 – Exemple de schéma d'analyse d'exécution de la requête

2.5 Exemple de journaux des réponses

comme mentionné plus tôt, les deux types de requêtes les plus utilisés sont les requêtes GET et les requêtes POST. Lorsqu'on accède à une page web nous utilisons la requête GET qui comprend des paramètres d'états. En outre, il est possible de déterminer les programmes utilisés par le service

web. D'ailleurs, des outils en ligne gratuits existent. Plus concrètement, la réponse envoyée par le serveur comprend beaucoup d'informations.

Pour la requête de type GET, il suffit d'inspecter la page. Par exemple, il est possible de savoir si un site web est conçu avec Wordpress en ne recherchant le préfix "wp-". La figure 2-4 suivante montre 76 résultats pour le préfix "wp-". Donc, la probabilité que ce site soit écrit avec Wordpress est très élevée.

FIGURE 2-4 – exemple d'analyse de la réponse GET

À noter qu'il est tout à fait possible d'analyser les requêtes d'autres types. Par exemple, on peut regarder l'en-tête de réponse pour vérifier si Cloudflare est utilisé comme proxy. La figure refcloudflare montre un exemple de réponse pour un type de requête POST.

En-têtes de réponse

Nom	Valeur
Content-Encoding	br
Cache-Control	no-cache, must-revalidate, max-age=0
Access-Control-Allow-Origin	https://route.algosol.ca
Referrer-Policy	strict-origin-when-cross-origin
Vary	Accept-Encoding
Date	Wed, 09 Jun 2021 13:15:20 GMT
Expires	Wed, 11 Jan 1984 05:00:00 GMT
Access-Control-Allow-Credentials	true
Content-Type	text/html; charset=UTF-8
X-Frame-Options	SAMEORIGIN
X-Content-Type-Options	nosniff
Server	cloudflare
x-robots-tag	noindex

FIGURE 2-5 – Exemple d'analyse de la réponse POST

3 Relation entre programmation, journaux et identification

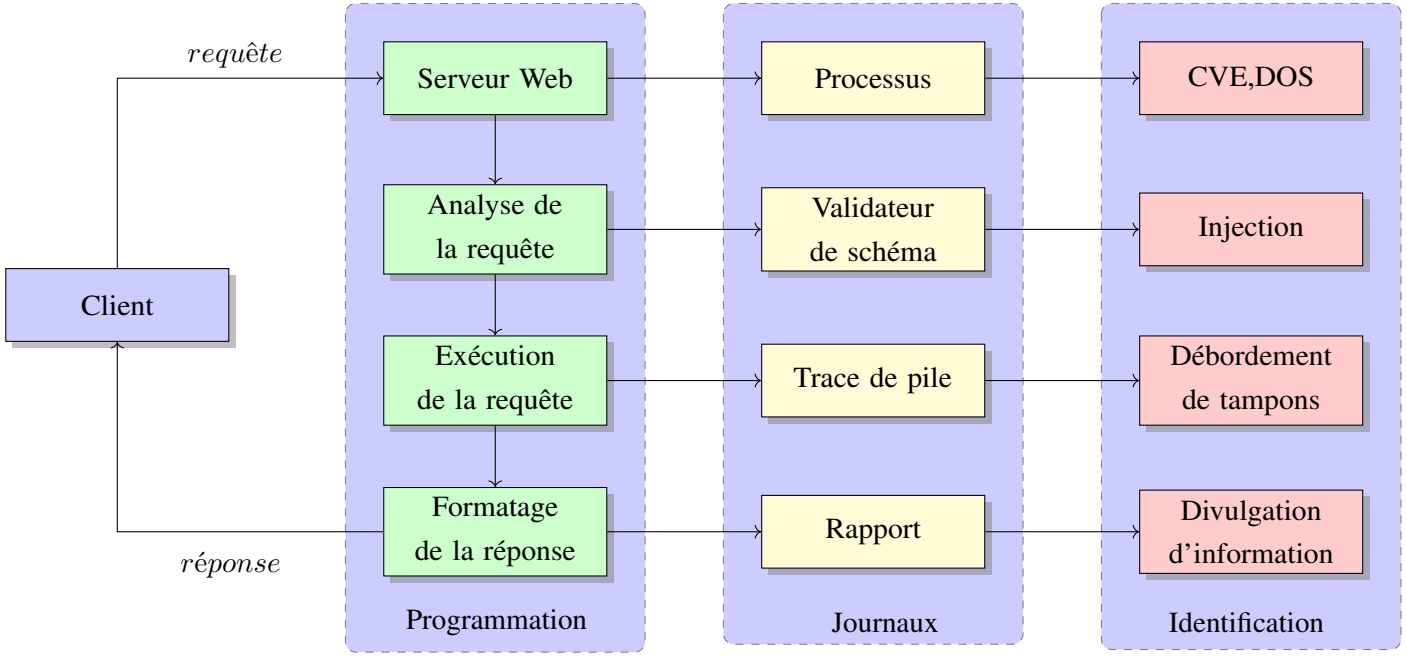

FIGURE 3-1 – Exemple de schéma d'analyse sécuritaire des journaux

4 Conclusion

cette publication rassemble des concepts d'analyse fondamentaux pour la protection des systèmes informatiques. En effet, la journalisation des événements est une mine d'or d'informations sur les tentatives d'intrusions et de mauvaises utilisations des systèmes informatiques. Cette publication n'est qu'une introduction pour l'analyse d'événements des journaux, en effet il existe des moyens plus efficaces d'analyser automatiquement les événements des journaux. Connaître les techniques offensives est le meilleur moyen de se prémunir des attaques informatiques.