LOG6302A — Analyse d'applications et Cyber-sécurité

Laboratoire #5

Donné le : Mar 21 2024, 09 :30 AM
Échéance : Apr 04 2024, 09 :30 AM

- Il s'agit d'un travail en équipe de deux.
- Chaque groupe doit rendre sur Moodle une archive contenant leur code et un rapport (PDF) avant la date limite.
- Le rapport doit rendre compte de ce que vous avez fait et les problèmes rencontrés. Vous pouvez discuter de tout autre élément que vous jugeriez pertinent.
- Chaque jour de retard entraîne une pénalité de 50%.
- Si vous avez des questions, vous pouvez demander des clarifications sur Discord (#lab-question)

Analyse de teinte

Dans ce laboratoire, nous allons implémenter une analyse de teinte permettant de s'assurer que toutes données venant de l'extérieure (entrées utilisateur) n'atteignent pas directement les points critiques du programme.
Nous allons dans un premier temps créer cette analyse et la tester sur des fichiers de test PHP. Ensuite, nous allons analyser un site web et y chercher de potentiel vulnérabilité d'injection SQL.

1 Implémentation et tests

Implémenter l'algorithme d'analyse de teinte vue en cours (*moodle - 3 :Définitions Possiblement Teintées*), et tester votre implémentation sur les quatre exemples suivant (*dossier part_1*). Le dataflow *(def, ref)* et les nœuds '*sources*', '*safe*', '*filters*' et '*sinks*' **sont fournis dans les fichiers .taint.json**.

```php
$clean = 'data';
$tainted = $_GET['data'];

sink($clean);
sink($tainted);
```

(a) File 1

```php
$clean = 'data';
$tainted = $_GET['data'];
$tainted2='data'.$tainted.$clean;
$clean2 = 'data' . $clean;

sink($clean2);
sink($tainted2);
```

(b) File 2

```php
$tainted = $_GET['data'];
$clean = filter($tainted);

sink($clean);
```

(c) File 3

```php
$tainted = $_GET['data'];
if($condition)
    $tainted = filter($tainted);
sink($tainted);
```

(d) File 4

2 Analyser une plateforme web

2.1 Mise en situation

Une entreprise du milieu médical vous demande d'effectuer une analyse en "boite blanche" sur leur service web. Et notamment de vérifier que leur application web est bien protégée contre les attaques d'injection SQL. L'architecture du système est décrite dans la figure suivante (fig. 2).

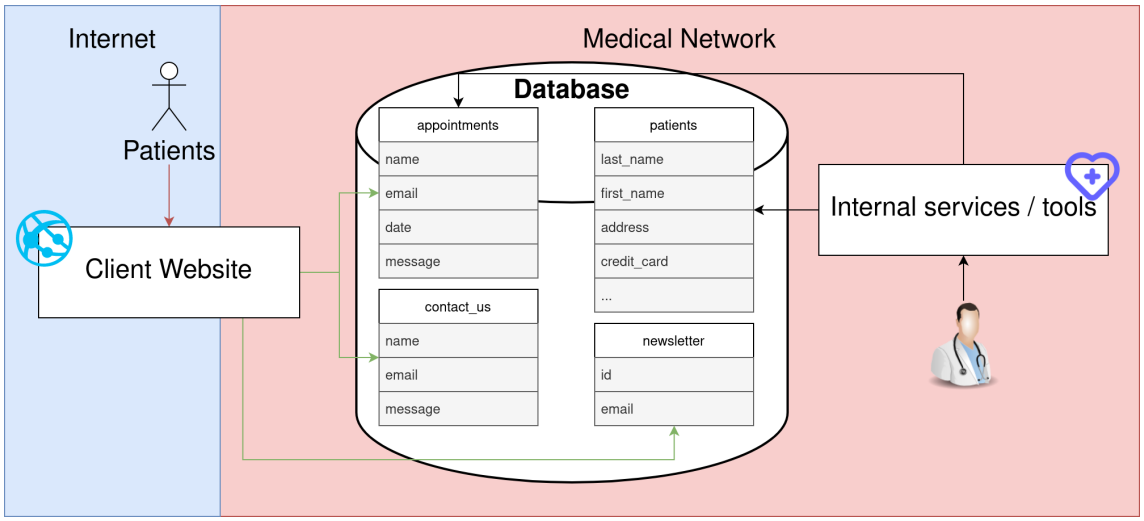

FIGURE 2 – Architecture

Vous devez effectuer une analyse de teinte sur la partie identifié "Client Website", pour vérifier que toutes les données provenant de l'extérieur (flèche rouge), n'atteignent pas directement la base de données sans être filtré (flèches vertes), ce qui pourrait mener à une faille d'injection SQL.

2.2 Détail de l'analyse

La plateforme n'utilise pas de *framework* particulier, et utilise l'extension PHP *mysqli* pour les interactions avec la base de données. Pour l'analyse de teinte nous considèrerons donc les ensembles suivants (tab. 1), fournis dans les fichiers *.taint.json*.

Ensembles	Éléments PHP

SOURCES	`$_POST`
	`$_GET`
	`$_REQUEST`
	`$_COOKIE`

SINKS	`$conn->query($sink);`
FILTERS	`filter_var($var);`
SAFE_SET	All literal values

TABLE 1 – Détails

2.3 Plateforme de test

Vous trouverez le code source à analyser sur Moodle (*dossier part_2*), et une version *'live'* du site web accessible par les liens ci-dessous (tab. 2). Si l'analyse de teinte révèle une vulnérabilité, essayez de l'exploiter pour démontrer sa présence.

Groupe	Lien
1	http://log6302a.info.polymtl.ca:8065/
2	http://log6302a.info.polymtl.ca:8288/
3	http://log6302a.info.polymtl.ca:8530/
4	http://log6302a.info.polymtl.ca:8856/
5	http://log6302a.info.polymtl.ca:8723/
6	http://log6302a.info.polymtl.ca:8828/
7	http://log6302a.info.polymtl.ca:8404/
8	http://log6302a.info.polymtl.ca:8609/

TABLE 2 – Plateformes

Question bonus Est-il possible d'extraire de la base de données des informations confidentielles relatives aux patients ?

3 Rapport

Détailler dans le rapport les résultats de votre analyse, incluant les IN / OUT de chaque nœuds, et justifier l'identification des *sinks* vulnérable obtenu. Discuter ensuite des limites de l'analyse, notamment liées aux ensembles choisi précédemment (tab. 1).