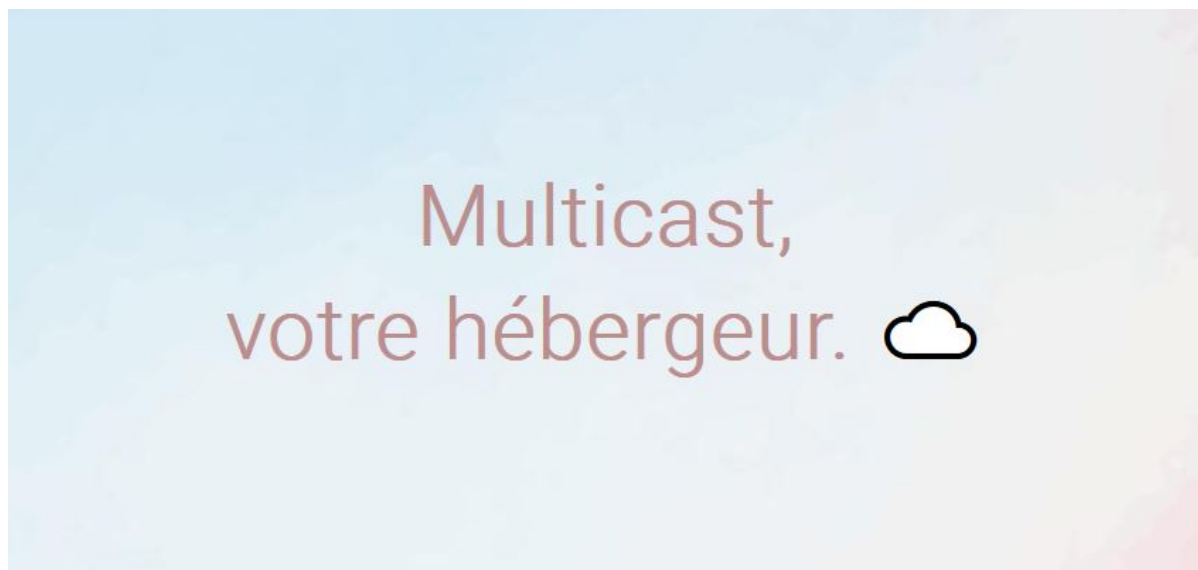


Tests et mesures de sécurité de l'application web '**Multicast**'



Scanners automatiques :

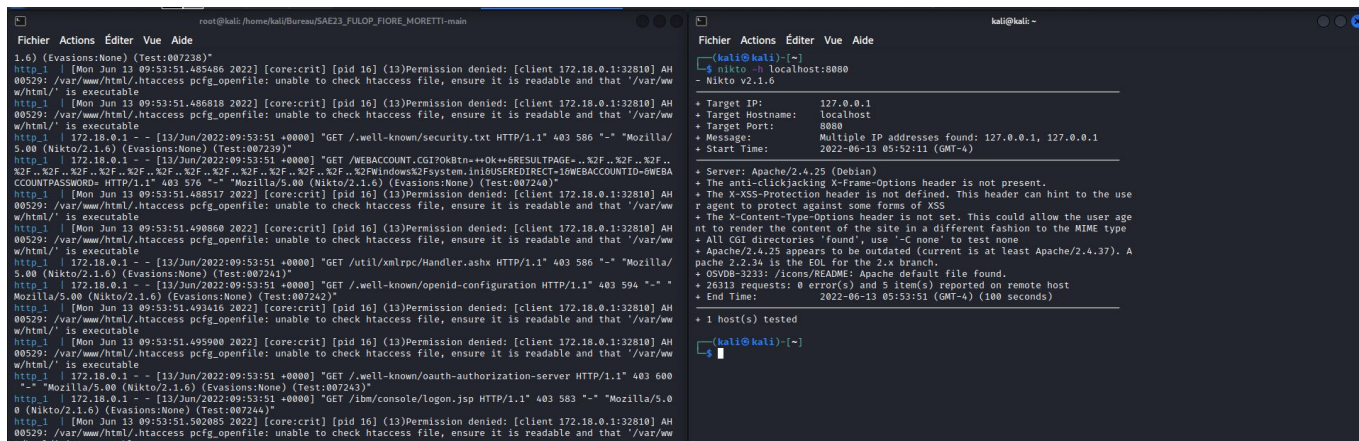
Pour chercher les failles les plus récurrentes et rapidement et surtout pour nous donner un plan d'ensemble des vulnérabilités présentes sur notre site, nous avons utilisé des outils de scans automatiques.

Nikto :

Nikto est un scanner de vulnérabilité en ligne de commande logiciel gratuit qui analyse les serveurs Web à la recherche de fichiers/CGI dangereux, de logiciels serveur obsolètes et d'autres problèmes. Il effectue des vérifications génériques et spécifiques au type de serveur.

Commande lancé :

```
nikto -h localhost:8080
```



Retour du test :

- Version de Apache pas mise à jour (2.4.25 au lieu de 2.2.34) ~ Cependant, pas de failles majeures sur cette version-ci.
- Pas de présence de protection anti-clickjacking dans le header du serveur apache :

~ Résolution du problème dans le fichier de configuration /etc/apache2/conf-enabled/security.conf sur apache :

```
a2enmod headers # Ajout d'un module
```

```
sudo nano 000-default.conf # Modification configuration
```

```
Header set X-Frame-Options "DENY"
```

```
Header set Content-Security-Policy "frame-ancestors 'none'"
```

Nessus :

Nessus est un outil de sécurité informatique. Il signale les faiblesses potentielles ou avérées sur les machines testées. Ceci inclut, entre autres : les services vulnérables à des attaques permettant la prise de contrôle de la machine, l'accès à des informations sensibles, des dénis de service...

Test effectué : **Web Application Test**

test final

Configure Audit Trail Launch Report Export

Hosts 1 Vulnerabilities 17 Notes 2 VPR Top Threats History 2

Filter Search Vulnerabilities 17 Vulnerabilities

| Sev | Score | Name | Family | Count |
|--------|-------|--|---------------|-------|
| MEDIUM | 4.3 * | Web Application Potentially Vulnerable to Clickjacking | Web Servers | 1 |
| MIXED | ... | Web Server (Multiple Issues) | Web Servers | 6 |
| INFO | ... | HTTP (Multiple Issues) | Web Servers | 10 |
| INFO | ... | HTTP (Multiple Issues) | CGI abuses | 3 |
| INFO | ... | Netstat Portscanner (SSH) | Port scanners | 5 |
| INFO | ... | Web Application Cookies Not Marked Secure | Web Servers | 4 |
| INFO | ... | Apache HTTP Server Version | Web Servers | 2 |
| INFO | ... | Web Application Sitemap | Web Servers | 2 |
| INFO | ... | CGI Generic Injectable Parameter | CGI abuses | 1 |
| INFO | ... | CGI Generic Tests Load Estimation (all tests) | CGI abuses | 1 |
| INFO | ... | CGI Generic Tests Timeout | CGI abuses | 1 |
| INFO | ... | External URLs | Web Servers | 1 |
| INFO | ... | jQuery Detection | CGI abuses | 1 |

Scan Details

Policy: Web Application Tests
 Status: Completed
 Severity Base: CVSS v3.0
 Scanner: Local Scanner
 Start: Today at 4:51 AM
 End: Today at 5:28 AM
 Elapsed: 37 minutes

Vulnerabilities

Retour du test :

- Application vulnérable au Clickjacking -> Cependant résolu après retour du test précédent
- Aucunes failles majeures ou hautes sur le site détecté
- Transmission des informations en clair (expliqué par le fait qu'on n'utilise pas d'https)

Résumé des tests :

- Aucune faille majeure n'a été détecté parmi les deux tests automatiques.
- Une faille moyenne a été résolue.

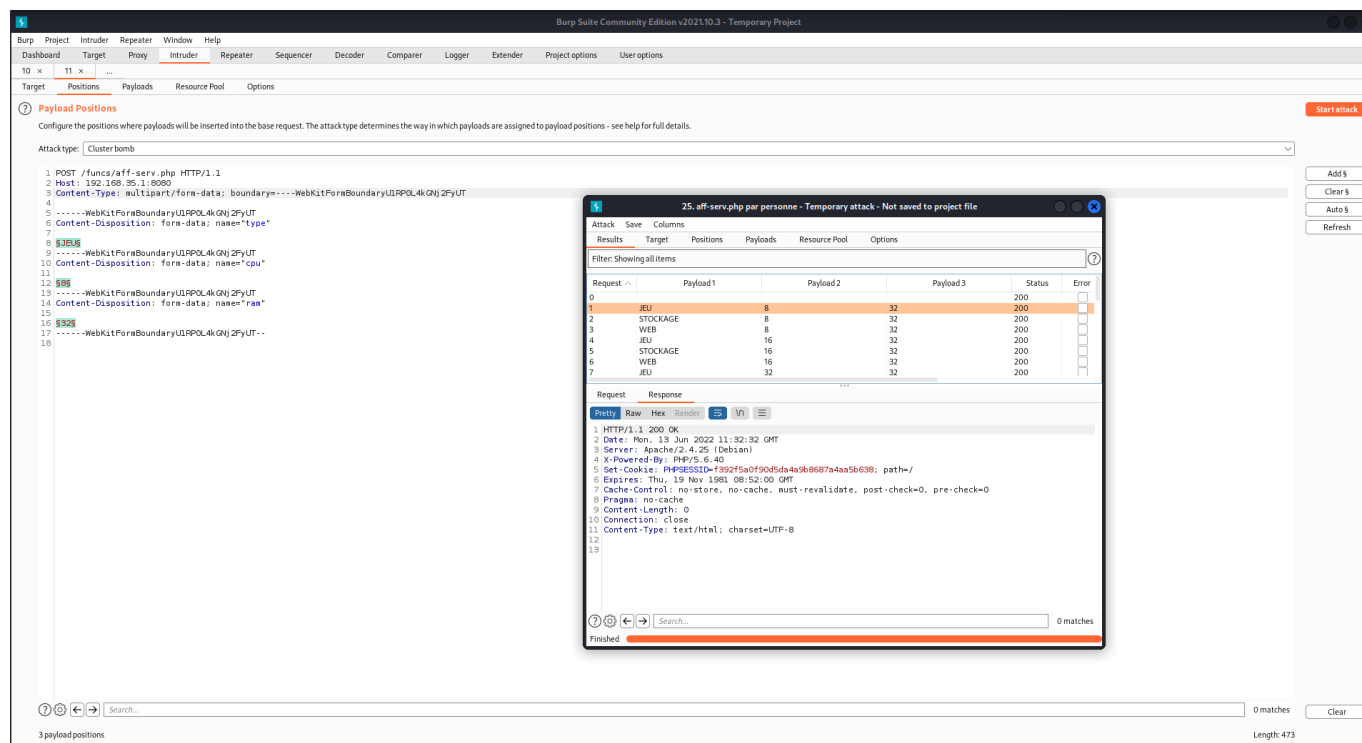
Tests manuels (Burpsuite / Injections Sql) :

Parce-que les scanners automatiques ne sont pas toujours fiables, nous avons tester certaines actions que ne devraient normalement pas êtres autorisés pour un utilisateur lambda ou sans compte.

Tests sur la fonction 'Aff-serv.php' :

Cette fonction retourne un tableau JSON selon ce qu'à demander l'utilisateur (TYPE, RAM, CPU). **Cette fonction est utilisée sur la page 'Location' et doit être accessible seulement aux utilisateurs connectés**

Premier test en tant qu'utilisateur non connecté :



The screenshot shows the Burp Suite interface with a cluster bomb attack configured on the 'Aff-serv.php' endpoint. The attack is named '25. aff-serv.php par personne - Temporary attack - Not saved to project file'. The results table shows that all three payloads were successfully sent to the server with a status of 200. The response is displayed in the 'Response' tab, showing an HTTP 200 OK status and a 'Set-Cookie' header.

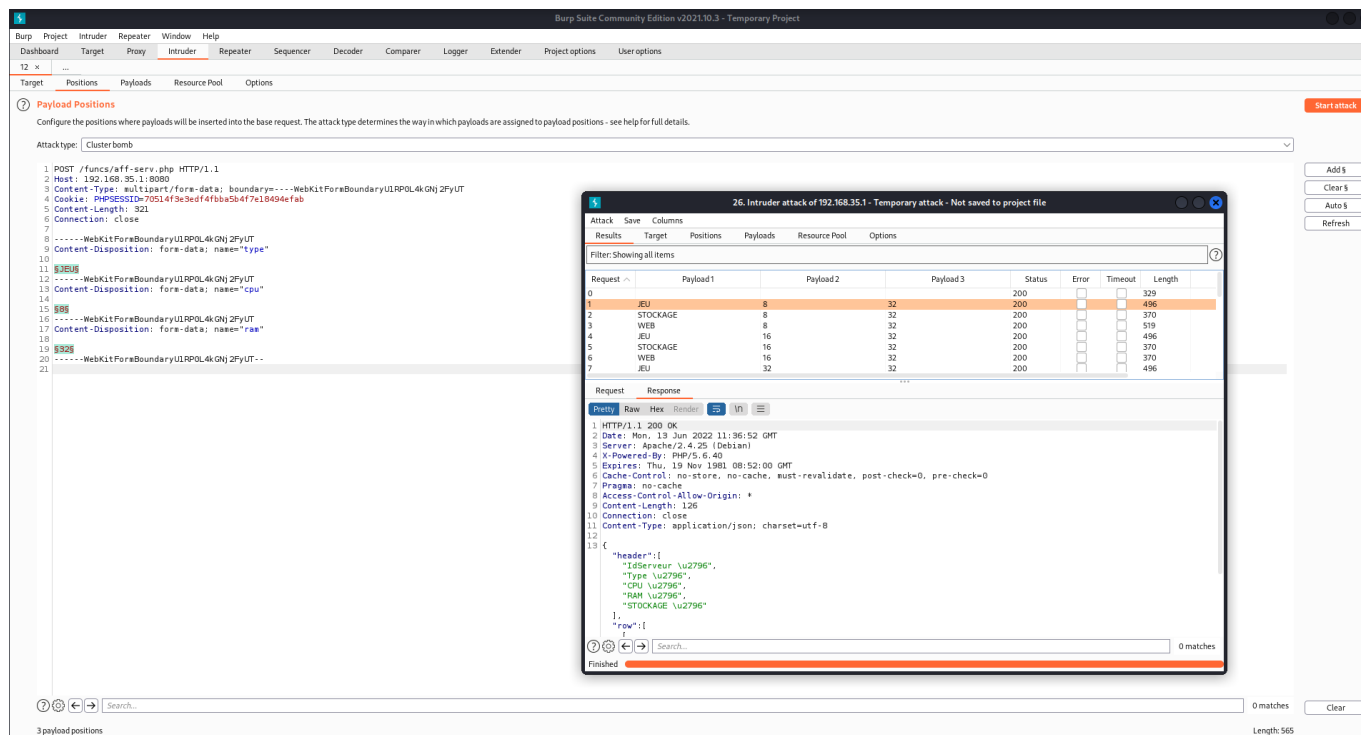
| Request | Payload1 | Payload2 | Payload3 | Status | Error |
|---------|----------|----------|----------|--------|-------|
| 1 | JEU | 8 | 32 | 200 | |
| 2 | STOCKAGE | 8 | 32 | 200 | |
| 3 | WEB | 8 | 32 | 200 | |
| 4 | JEU | 16 | 32 | 200 | |
| 5 | STOCKAGE | 16 | 32 | 200 | |
| 6 | WEB | 16 | 32 | 200 | |
| 7 | JEU | 32 | 32 | 200 | |

Response:

```
1 HTTP/1.1 200 OK
2 Date: Mon, 13 Jun 2022 11:32:32 GMT
3 Server: Apache/2.4.25 (Debian)
4 X-Powered-By: PHP/5.6.40
5 Set-Cookie: PHPSESSID=f992f5a0f90d5da4a9b8687a4aa5b638; path=/
6 Expires: Thu, 19 Nov 1981 08:52:00 GMT
7 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
8 Pragma: no-cache
9 Content-Length: 0
10 Connection: close
11 Content-Type: text/html; charset=UTF-8
12
13
```

On remarque qu'il n'y a aucune réponse en body, ce qui veut dire que sans cookie de connexion, la page ne nous retourne rien.

Deuxième test en tant qu'utilisateur connecté :



The screenshot shows the Burp Suite Community Edition v2021.10.3 interface. The main window displays the 'Payload Positions' tab, which is used to configure the positions where payloads will be inserted into the base request. The attack type is set to 'Cluster bomb'. The base request is a POST to /funcs/aff-serv.php HTTP/1.1. The request body contains a multipart/form-data boundary with several fields: 'name' (type), 'cpu', 'ram', and 'stockage'. The 'Intruder' tab is also visible, showing the attack configuration. The 'Results' tab shows a table of attack results. The table has columns for Request, Payload1, Payload2, Payload3, Status, Error, Timeout, and Length. The results show that the attack was successful, with a status of 200 and a length of 496. The 'Payload Positions' tab shows the configuration for the attack, including the target URL and the positions where payloads will be inserted.

| Request | Payload1 | Payload2 | Payload3 | Status | Error | Timeout | Length |
|---------|----------|----------|----------|--------|-------|---------|--------|
| 0 | JEU | 8 | 32 | 200 | | | 329 |
| 1 | JEU | 8 | 32 | 200 | | | 496 |
| 2 | STOCKAGE | 8 | 32 | 200 | | | 370 |
| 3 | JEU | 8 | 32 | 200 | | | 519 |
| 4 | JEU | 16 | 32 | 200 | | | 496 |
| 5 | STOCKAGE | 16 | 32 | 200 | | | 370 |
| 6 | WEB | 16 | 32 | 200 | | | 496 |
| 7 | JEU | 32 | 32 | 200 | | | 496 |

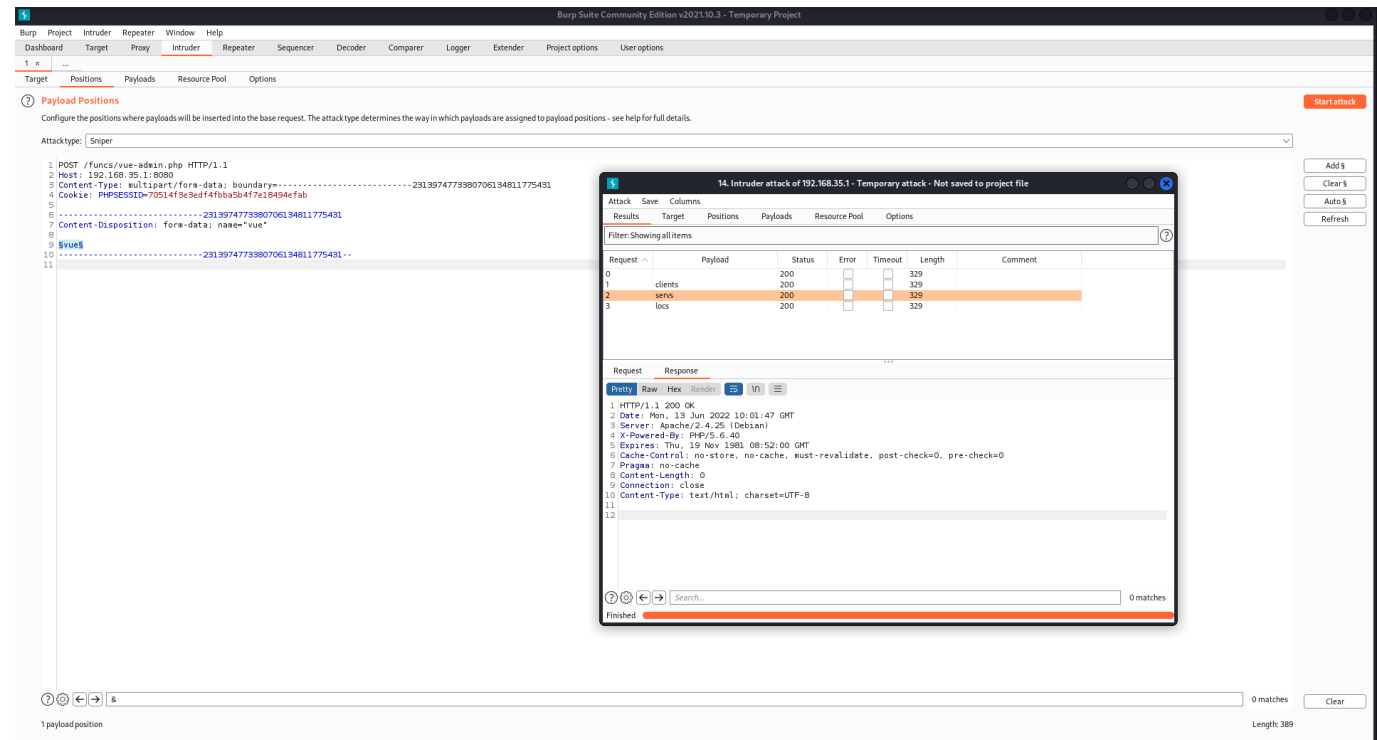
Avec le cookie de connexion d'un utilisateur lambda, nous avons bien en retour une réponse (en vert en partie), ce que nous n'avions pas auparavant. La fonction fonctionne donc bien seulement avec un utilisateur connecté.

Test passé avec succès !

Tests sur la fonction 'vue-admin.php' :

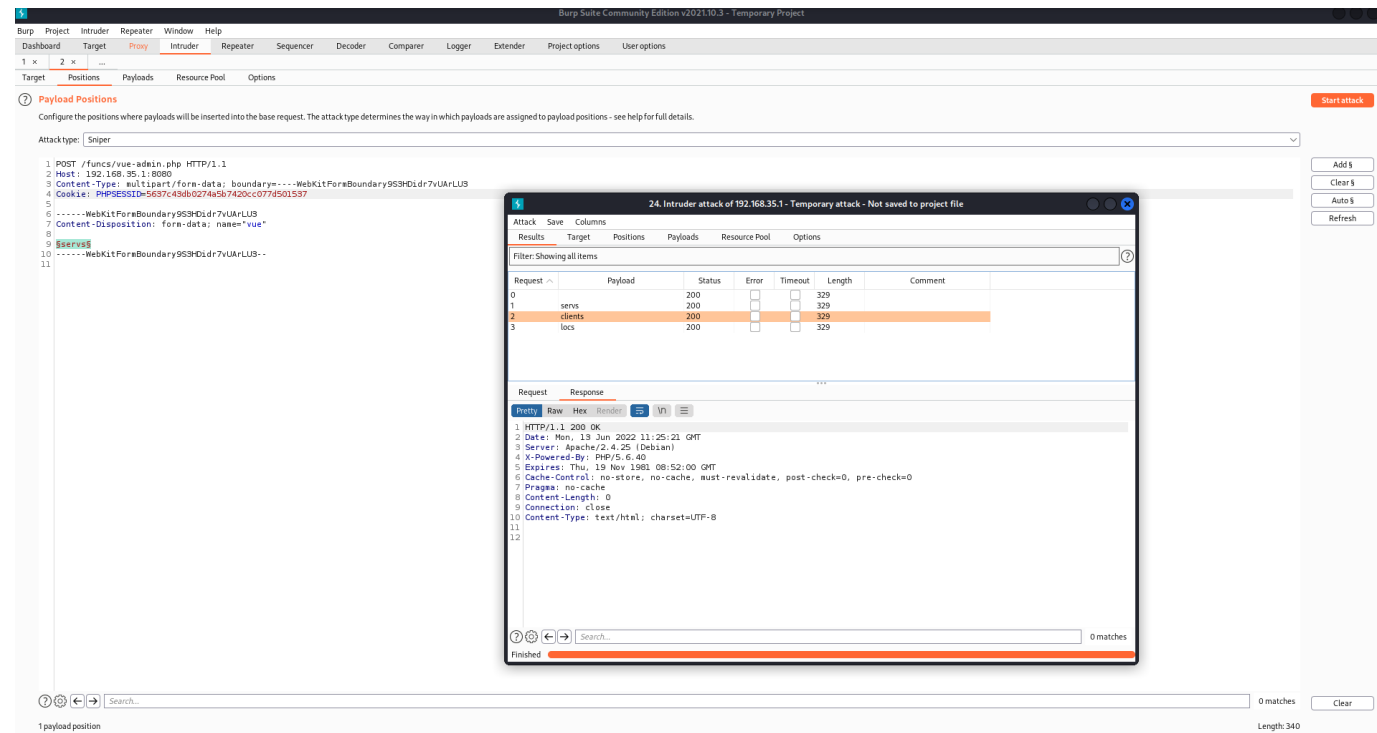
Cette fonction retourne des données json qui constituent la page pannel d'administration destiné à l'administration des serveurs. **Cette fonction est utilisée sur la page 'gestion' et doit être accessible seulement à l'utilisateur admin.**

Premier test en tant qu'utilisateur non connecté :



On remarque qu'il n'y a aucune réponse en body, ce qui veut dire que sans cookie de connexion, la page ne nous retourne rien comme prévu.

Deuxième test en tant qu'utilisateur connecté :



Avec le cookie de connexion d'un utilisateur lambda, nous avons bien également zéro retour de la part de la fonction.

Troisième test en tant qu'utilisateur admin :

The screenshot shows the Burp Suite interface with an intruder attack configured. The 'Attack type' is set to 'Sniper'. The 'Payloads' tab is active, showing a list of payloads with columns for Request, Payload, Status, Error, Timeout, Length, and Comment. The results show three payloads: 'clients', 'seme', and 'locs', all with a status of 200. The 'Request' tab is also visible, showing the HTTP request details.

| Request | Payload | Status | Error | Timeout | Length | Comment |
|---------|---------|--------|-------|---------|--------|---------|
| 0 | | 200 | | | 329 | |
| 1 | clients | 200 | | | 609 | |
| 2 | seme | 200 | | | 793 | |
| 3 | locs | 200 | | | 706 | |

Avec le cookie de connexion de l'administrateur du site, nous avons bien des données reçues de la part de la fonction réservé à l'admin. Nos données sont donc bien sécurisées face aux requêtes malveillantes !

Test passé avec succès !

Au niveau des injections SQL / XSS :

Nous avons sécurisé nos entrées utilisateurs grâce à des fonctions php. Chaque entrée est 'sanitized' grâce aux fonction htmlspecialchars contre les failles XSS ainsi que pdo.prepare contre les injections SQL.

Cependant, si la fonction PDO prepare n'était pas possible à intégrer, on vérifie l'entrée utilisateur en faisant attention qu'elle rentre bien dans un dictionnaire que nous avons prédéfini à l'avance pour ne pas qu'il puisse mettre ce qu'il veut à la place.