École Polytechnique de Montréal

Département de génie informatique et génie logiciel

Rapport du TP2

Cybersécurité

Travail pratique 2

Fares Laadjel, 2297799

Julien Cyr, 2278776

Table des matières

1 Reconnaissance [/3] 2

2 Accès initial [/7] 3
2.1 Injection SQL sur la page de connexion [/1] 3
2.2 Connexion par injection SQL [/2] 4
2.3 Injection XSS sur la page de tickets [/1] 4
2.4 Récupération des cookies admin par XSS [/2] 5
2.5 Connexion avec le cookie récupéré [/1] 6

3 Analyse en boîte blanche [/5] 7
3.1 Récupération du code source via FTP [/1] 7
3.2 Correction de la vulnérabilité FTP [/1] 9
3.3 Correction de l'injection SQL dans le code PHP [/1] 9
3.4 Mécanisme de basculement des services [/2] 10

4 Compromission d'un compte utilisateur [/4] 10
4.1 Récupération du hash de carol par injection SQL [/2] 10
4.2 Craquage du hash [/1] 10
4.3 Récupération de controller.c via SSH [/1] 11

5 Élévation de privilèges [/6] 11
5.1 Shell www-data par injection de commande PHP [/2] 11
5.2 Analyse de controller.c et vulnérabilité buffer overflow [/2] 11
5.3 Exploitation du buffer overflow pour devenir root [/2] 11

6 Accès physique = Game Over [3 points bonus] 11

Bibliographie 12

1 Reconnaissance [/3]

Utilisez `nmap` pour scanner le serveur situé en 10.22.0.11. Identifiez les services et le système d'exploitation de la machine. Indiquez les captures d'écran des commandes utilisées et de leur sortie. Plus tard dans le TP, mettez à jour la liste si de nouveaux services apparaissent.

Après avoir lancé la machine Kali et effectué un scan avec nmap sur 10.22.0.11 en ignorant le port 2222, deux services sont actuellement exposés.

Le port 22/tcp est ouvert et exécute le service SSH. La version détectée est OpenSSH 8.9p1 sur Ubuntu. La machine permet des connexions distantes sécurisées et fonctionne sur une distribution Linux de type Ubuntu.

Le port 8080/tcp est ouvert et exécute un serveur web Apache 2.4.52 sur Ubuntu. Le scan révèle la présence d'une interface web menant vers une page de connexion login.php, ce qui suggère une application web avec authentification.

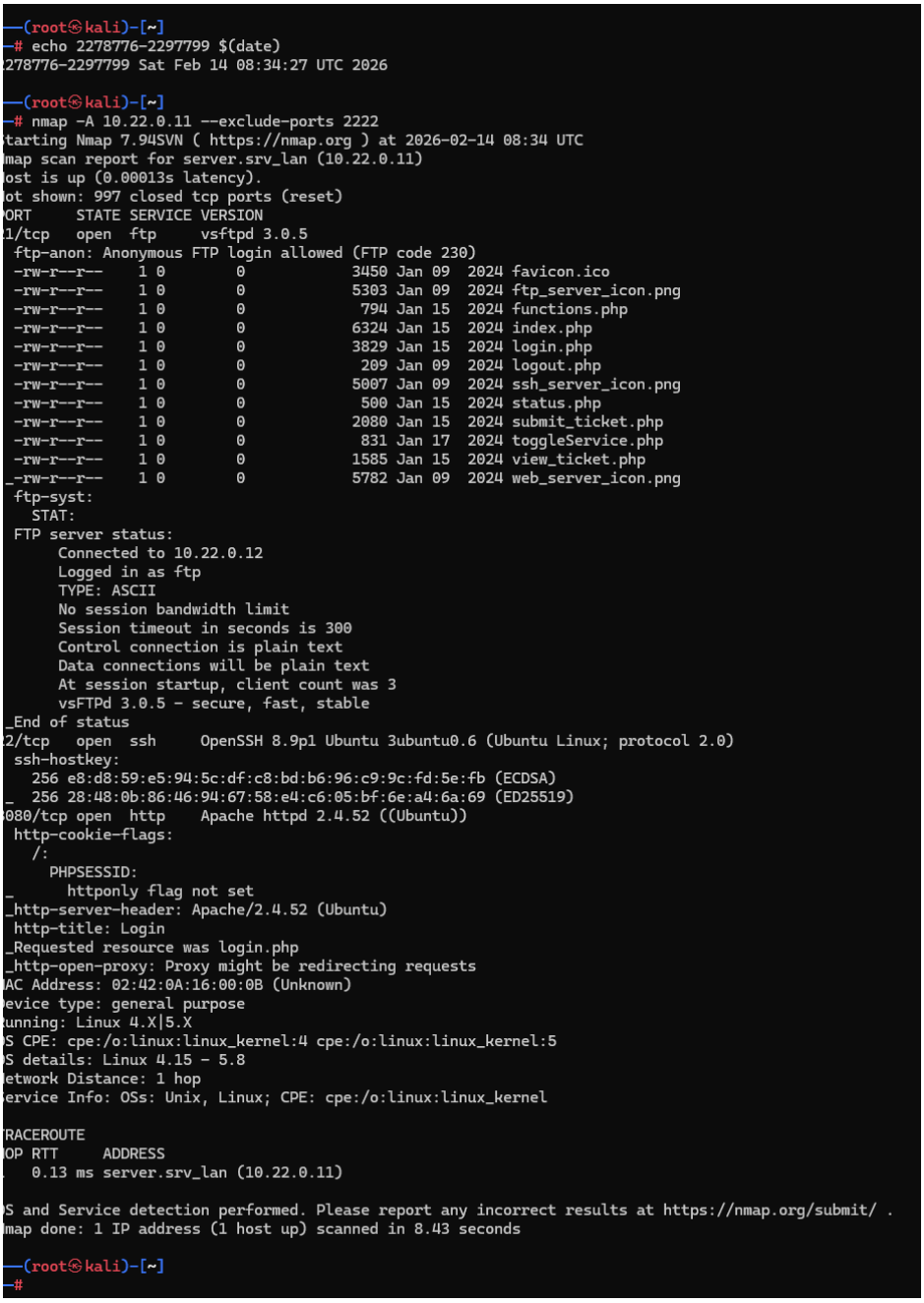
La détection du système d'exploitation indique un noyau Linux entre les versions 4.15 et 5.8. En combinant cette information avec les bannières des services, on peut conclure que la machine exécute probablement Ubuntu Linux.

```
┌──(root㉿kali)-[~]
└─# nmap -A 10.22.0.11 --exclude-ports 2222
Starting Nmap 7.94SVN ( https://nmap.org ) at 2026-02-14 04:26 UTC
Nmap scan report for server.srv_lan (10.22.0.11)
Host is up (0.000083s latency).
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 e8:d8:59:e5:94:5c:df:c8:bd:b6:96:c9:9c:fd:5e:fb (ECDSA)
|_  256 28:48:0b:86:46:94:67:58:e4:c6:05:bf:6e:a4:6a:69 (ED25519)
8080/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
|_http-open-proxy: Proxy might be redirecting requests
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-title: Login
|_Requested resource was login.php
|_http-server-header: Apache/2.4.52 (Ubuntu)
MAC Address: 02:42:0A:16:00:0B (Unknown)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.08 ms server.srv_lan (10.22.0.11)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.22 seconds

┌──(root㉿kali)-[~]
└─# echo 2278776-2297799 $(date)
2278776-2297799 Sat Feb 14 04:38:09 UTC 2026

┌──(root㉿kali)-[~]
└─#
```

Après l'activation du service FTP depuis l'interface web, un nouveau scan nmap de 10.22.0.11 montre l'apparition d'un service supplémentaire sur le port 21. Le service détecté est un serveur FTP vsftpd en version 3.0.5.

Le scan indique aussi que l'accès FTP anonyme est autorisé. Une connexion sans identifiants permet de lister directement le contenu exposé. La liste retournée contient plusieurs fichiers du site, notamment index.php, login.php, functions.php, toggleService.php, submit_ticket.php et view_ticket.php. Cela confirme que le service FTP donne accès à des fichiers du code source de l'application.

La liste des services observés à ce stade comprend donc un service FTP sur le port 21, un service SSH sur le port 22 et un service web HTTP sur le port 8080.



2 Accès initial [/7]

2.1 Injection SQL sur la page de connexion [/1]

Montrez que la page de connexion est vulnérable aux injections SQL.

En entrant simplement une quote dans le champ utilisateur (par exemple admin'), le serveur retourne une erreur SQL affichant la requête complète exécutée. Cette erreur montre que l'entrée utilisateur est insérée directement dans la requête :

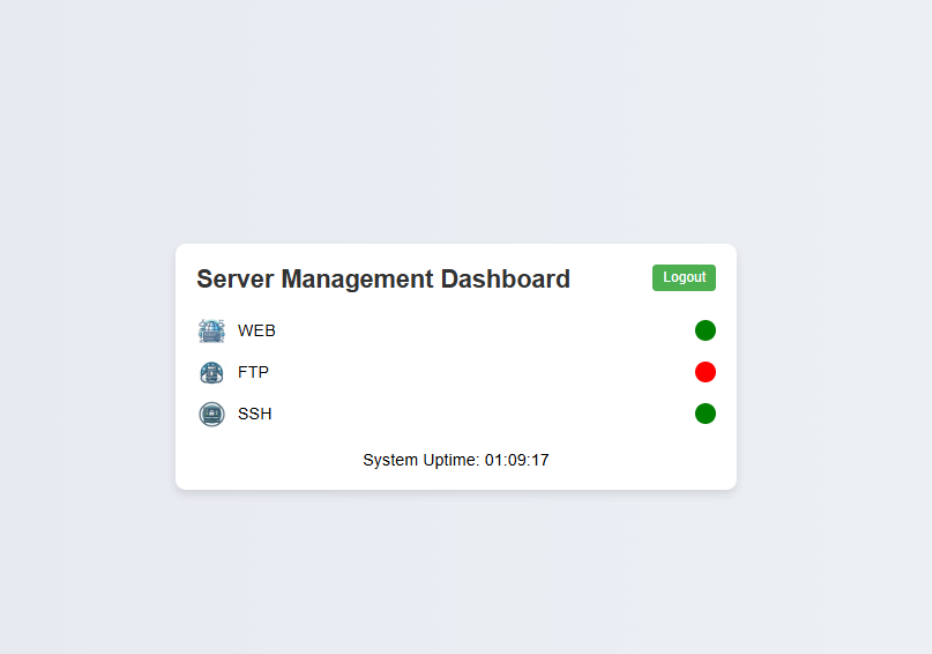
```
SELECT * FROM users WHERE username = 'admin" AND password =
'...'
```

La présence d'une erreur de syntaxe causée par l'entrée utilisateur prouve que les données ne sont pas filtrées ni échappées correctement. Comme le contenu saisi modifie la structure de la requête SQL et provoque une erreur côté base de données, cela démontre que la page de connexion est vulnérable aux injections SQL.

2.2 Connexion par injection SQL [/2]

En utilisant une injection SQL, connectez-vous sur le site. Expliquez avec vos mots comment fonctionne votre attaque.

La page de connexion est vulnérable aux injections SQL car les entrées utilisateur sont directement intégrées dans la requête SQL sans validation ni requêtes paramétrées. En entrant une chaîne spéciale contenant une quote et un commentaire SQL, il est possible de modifier la structure de la requête envoyée à la base de données. L'erreur SQL affichée par le serveur révèle la requête complète exécutée, ce qui prouve que l'entrée utilisateur est concaténée telle quelle dans le SQL. Cela permet de contourner la vérification du mot de passe et démontre une vulnérabilité d'injection SQL.



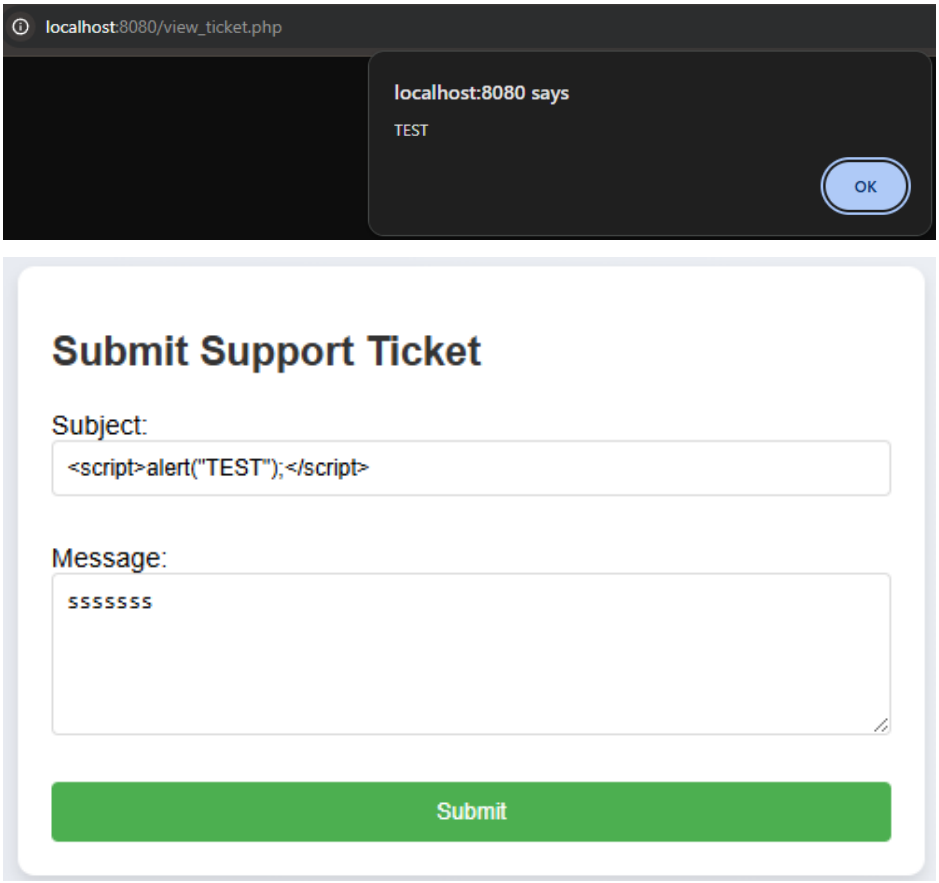
2.3 Injection XSS sur la page de tickets [/1]

Montrez que la page de soumission de tickets est vulnérable aux injections XSS.

Un ticket a été soumis avec le champ Subject contenant le script suivant :

```
<script>alert("TEST");</script>
```

Après soumission, l'ouverture du ticket déclenche l'exécution du script dans le navigateur et une boîte de dialogue affiche TEST. Cela démontre que l'entrée utilisateur est enregistrée par le serveur puis réaffichée sans être filtrée, et qu'elle est exécutée côté client. La page de soumission de tickets est donc vulnérable à une XSS stockée.



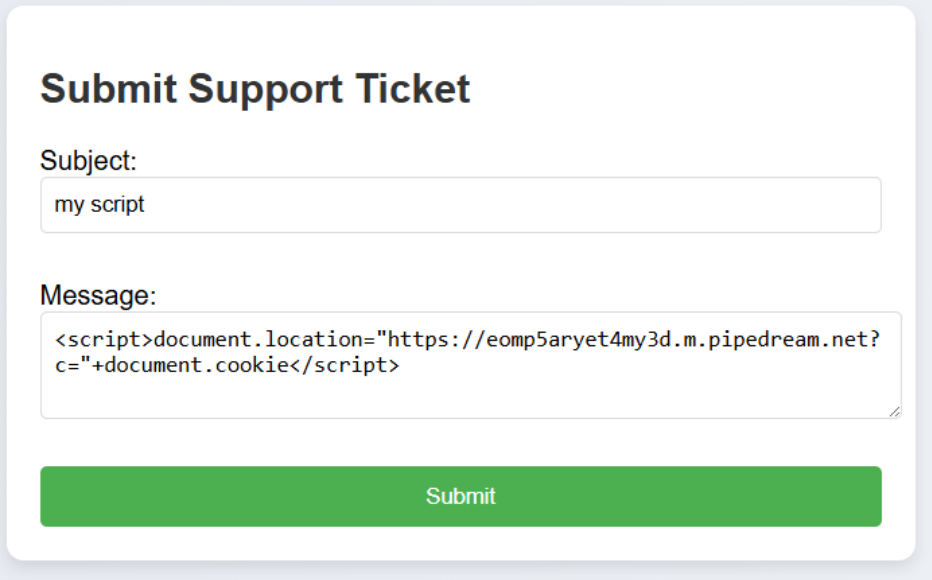
2.4 Récupération des cookies admin par XSS [/2]

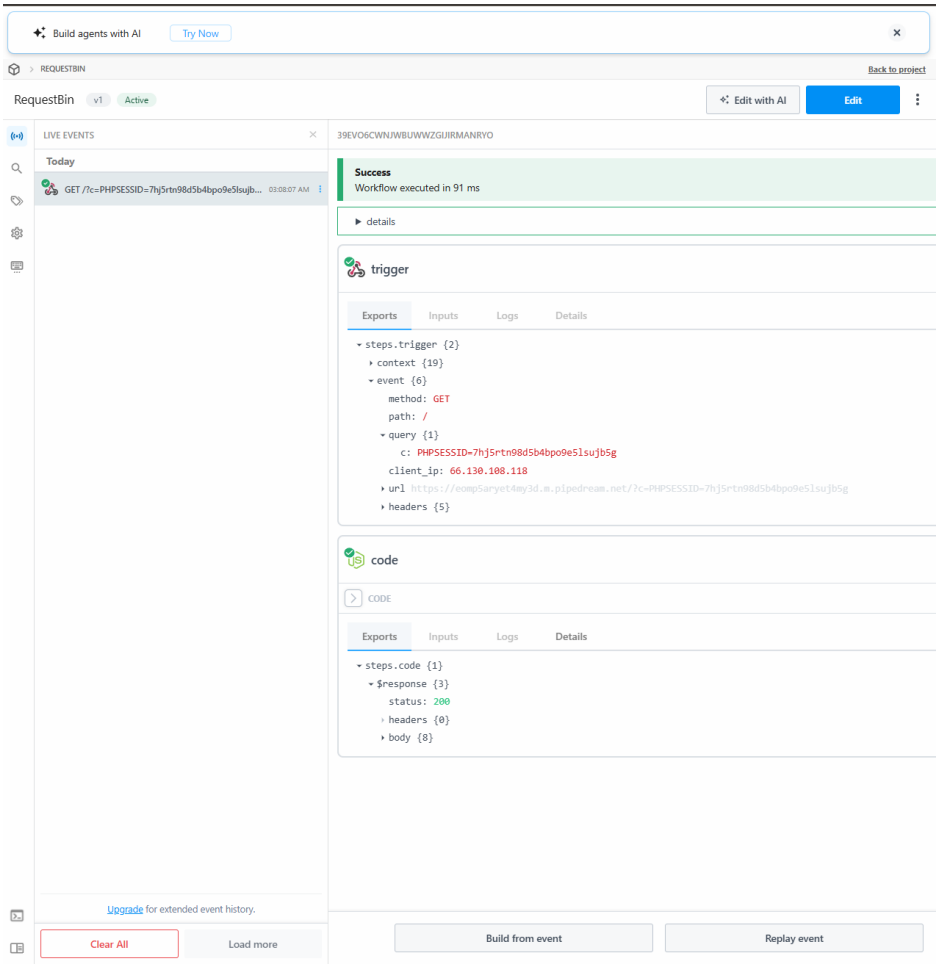
L'administratrice du système consulte régulièrement les tickets. Utilisez une injection XSS pour récupérer les cookies de son navigateur web.

La page de soumission de tickets accepte du contenu HTML et Javascript sans filtrage. Il est donc possible d'y injecter un script qui sera stocké sur le serveur puis exécuté dans le navigateur de toute personne qui consulte le ticket. Cette vulnérabilité correspond à une XSS persistante.

Un ticket contenant le script malveillant a été soumis. Le script envoie la valeur de document.cookie vers un endpoint externe contrôlé par l'attaquant. Lorsque l'administratrice ouvre la page des tickets, son navigateur exécute automatiquement le script dans le contexte du site. Le cookie de session de l'administratrice est alors transmis vers le serveur externe, ce qui permet de le récupérer sans interaction supplémentaire. La réception du cookie est confirmée par la requête observée sur le service RequestBin.

Cette attaque fonctionne parce que le site affiche le contenu des tickets sans validation ni échappement, ce qui permet l'exécution de code arbitraire dans le navigateur d'un utilisateur authentifié.





2.5 Connexion avec le cookie récupéré [/1]

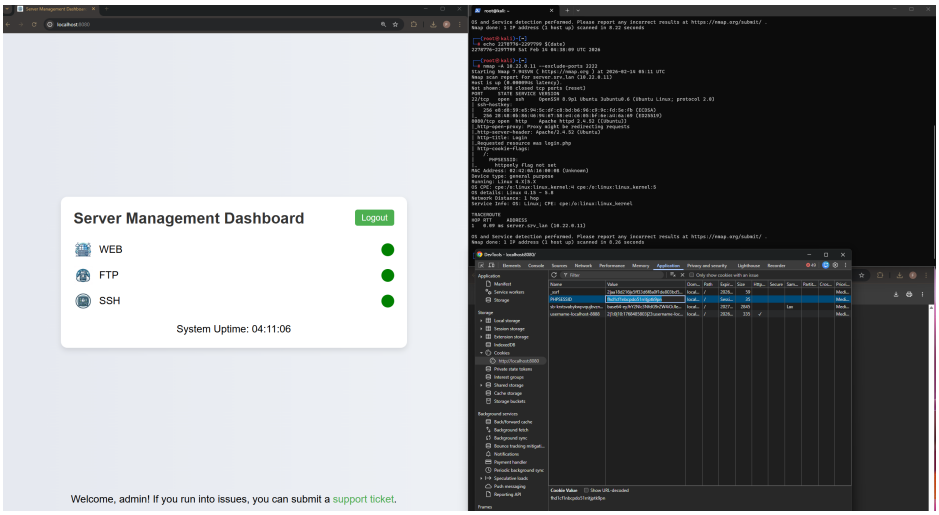
En utilisant le cookie récupéré, connectez-vous au site web en tant qu'admin.

Le cookie récupéré correspond à l'identifiant de session PHP de l'administratrice. Le mécanisme d'authentification du site repose uniquement sur ce cookie. Toute requête contenant un identifiant de session valide est considérée comme authentifiée par le serveur.

Le cookie de session local a été remplacé manuellement dans le navigateur par le

cookie volé. Après rechargement de la page, le serveur associe le navigateur à la session existante de l'administratrice. Il est alors possible d'accéder directement à l'interface administrateur sans connaître son mot de passe.

Cette étape démontre une usurpation de session. La possession du cookie suffit à prendre le contrôle du compte, ce qui montre l'impact critique de la vulnérabilité XSS exploitée à l'étape précédente.



3 Analyse en boîte blanche [/5]

Raccourci : Mot de passe admin : `iL0veTrains#78` si l'accès n'a pas été obtenu à la partie précédente.

3.1 Récupération du code source via FTP [/1]

Depuis l'application web, cliquez sur le bouton rouge pour allumer le service FTP. Connectez-vous au service FTP et récupérez le code source du site.

Une connexion FTP vers 10.22.0.11 a été établie en utilisant le nom d'utilisateur anonymous sans mot de passe. Le serveur vsftpd 3.0.5 accepte cette connexion et donne accès en lecture au répertoire contenant les fichiers de l'application web. La commande `ls` *permet de lister l'ensemble des fichiers exposés, incluant functions.php, index.php, login.php, logout.php, status.php, submit_ticket.php, toggleService.php et view_ticket.php, ainsi que des ressources statiques (icônes PNG).*

L'ensemble des fichiers a été récupéré à l'aide de la commande `mget *`*, qui télécharge tous les fichiers du répertoire courant. Le code source complet du site est ainsi obtenu localement sur la machine Kali.*

```
┌──(root㉿kali)-[~]
└─# ftp 10.22.0.11
Connected to 10.22.0.11.
220 (vsFTPd 3.0.5)
Name (10.22.0.11:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||11207|)
150 Here comes the directory listing.
-rw-r--r--    1 0        0            3450 Jan 09  2024 favicon.ico
-rw-r--r--    1 0        0            5303 Jan 09  2024 ftp_server_icon.png
-rw-r--r--    1 0        0             794 Jan 15  2024 functions.php
-rw-r--r--    1 0        0            6324 Jan 15  2024 index.php
-rw-r--r--    1 0        0            3829 Jan 15  2024 login.php
-rw-r--r--    1 0        0             209 Jan 09  2024 logout.php
-rw-r--r--    1 0        0            5007 Jan 09  2024 ssh_server_icon.png
-rw-r--r--    1 0        0             500 Jan 15  2024 status.php
-rw-r--r--    1 0        0            2080 Jan 15  2024 submit_ticket.php
-rw-r--r--    1 0        0             831 Jan 17  2024 toggleService.php
-rw-r--r--    1 0        0            1585 Jan 15  2024 view_ticket.php
-rw-r--r--    1 0        0            5782 Jan 09  2024 web_server_icon.png
226 Directory send OK.
```

```
ftp> mget *
mget favicon.ico [anpqy?]? y
229 Entering Extended Passive Mode (|||56378|)
150 Opening BINARY mode data connection for favicon.ico (3450 bytes).
100% |*****************************************************************************************|  3450        8.06 MiB/s    00:00 ETA
226 Transfer complete.
3450 bytes received in 00:00 (4.44 MiB/s)
mget ftp_server_icon.png [anpqy?]? y
229 Entering Extended Passive Mode (|||36378|)
150 Opening BINARY mode data connection for ftp_server_icon.png (5303 bytes).
100% |*****************************************************************************************|  5303      490.08 KiB/s    00:00 ETA
226 Transfer complete.
5303 bytes received in 00:00 (464.45 KiB/s)
mget functions.php [anpqy?]? y
229 Entering Extended Passive Mode (|||31407|)
150 Opening BINARY mode data connection for functions.php (794 bytes).
100% |*****************************************************************************************|   794        5.48 MiB/s    00:00 ETA
226 Transfer complete.
794 bytes received in 00:00 (2.67 MiB/s)
mget index.php [anpqy?]? y
229 Entering Extended Passive Mode (|||59053|)
150 Opening BINARY mode data connection for index.php (6324 bytes).
100% |*****************************************************************************************|  6324       30.92 MiB/s    00:00 ETA
226 Transfer complete.
6324 bytes received in 00:00 (11.46 MiB/s)
mget login.php [anpqy?]? y
229 Entering Extended Passive Mode (|||15140|)
150 Opening BINARY mode data connection for login.php (3829 bytes).
100% |*****************************************************************************************|  3829       33.81 MiB/s    00:00 ETA
226 Transfer complete.
3829 bytes received in 00:00 (6.56 MiB/s)
mget logout.php [anpqy?]? y
229 Entering Extended Passive Mode (|||63532|)
150 Opening BINARY mode data connection for logout.php (209 bytes).
100% |*****************************************************************************************|   209       85.11 KiB/s    00:00 ETA
226 Transfer complete.
209 bytes received in 00:00 (77.10 KiB/s)
mget ssh_server_icon.png [anpqy?]? y
229 Entering Extended Passive Mode (|||16829|)
150 Opening BINARY mode data connection for ssh_server_icon.png (5007 bytes).
100% |*****************************************************************************************|  5007        1.91 MiB/s    00:00 ETA
226 Transfer complete.
5007 bytes received in 00:00 (1.27 MiB/s)
mget status.php [anpqy?]? y
229 Entering Extended Passive Mode (|||41514|)
150 Opening BINARY mode data connection for status.php (500 bytes).
100% |*****************************************************************************************|   500        1.41 MiB/s    00:00 ETA
226 Transfer complete.
500 bytes received in 00:00 (334.89 KiB/s)
mget submit_ticket.php [anpqy?]? y
229 Entering Extended Passive Mode (|||14812|)
150 Opening BINARY mode data connection for submit_ticket.php (2080 bytes).
100% |*****************************************************************************************|  2080      202.65 KiB/s    00:00 ETA
226 Transfer complete.
2080 bytes received in 00:00 (196.95 KiB/s)
mget toggleService.php [anpqy?]? y
229 Entering Extended Passive Mode (|||46610|)
150 Opening BINARY mode data connection for toggleService.php (831 bytes).
100% |*****************************************************************************************|   831      744.51 KiB/s    00:00 ETA
226 Transfer complete.
831 bytes received in 00:00 (632.02 KiB/s)
mget view_ticket.php [anpqy?]? y
229 Entering Extended Passive Mode (|||27973|)
150 Opening BINARY mode data connection for view_ticket.php (1585 bytes).
100% |*****************************************************************************************|  1585        1.69 MiB/s    00:00 ETA
226 Transfer complete.
1585 bytes received in 00:00 (0.98 MiB/s)
mget web_server_icon.png [anpqy?]? y
229 Entering Extended Passive Mode (|||13576|)
150 Opening BINARY mode data connection for web_server_icon.png (5782 bytes).
100% |*****************************************************************************************|  5782      524.42 KiB/s    00:00 ETA
226 Transfer complete.
5782 bytes received in 00:00 (487.47 KiB/s)
ftp> y
?Invalid command.
ftp> exit
221 Goodbye.

┌──(root㉿kali)-[~]
└─# ls
favicon.ico          functions.php  login.php   ssh_server_icon.png  submit_ticket.php  view_ticket.php
ftp_server_icon.png  index.php      logout.php  status.php           toggleService.php  web_server_icon.png

┌──(root㉿kali)-[~]
└─# echo 2278776-2297799 $(date)
2278776-2297799 Sat Feb 14 08:49:45 UTC 2026
```

3.2 Correction de la vulnérabilité FTP [/1]

Expliquez précisément comment corriger la vulnérabilité qui permet cette connexion FTP.

La vulnérabilité provient du fait que le serveur FTP autorise les connexions anonymes. Toute personne sur le réseau peut donc se connecter sans identifiants et accéder directement aux fichiers exposés, incluant le code source de l'application. Cela facilite fortement l'analyse et l'exploitation de l'application.

La correction consiste à désactiver l'accès anonyme dans la configuration de vsftpd. Dans le fichier de configuration `/etc/vsftpd/vsftpd.conf`*, l'option* `anonymous_enable` *doit être réglée à* `NO`*. Après modification, le service FTP doit être redémarré pour appliquer la configuration.*

En complément, l'accès FTP devrait être limité aux seuls utilisateurs autorisés via des comptes authentifiés, avec des permissions minimales sur les répertoires. Le répertoire contenant le code source de l'application ne devrait jamais être exposé publiquement via FTP. Une solution plus sécuritaire consiste à remplacer FTP par SFTP ou un mécanisme de déploiement interne afin d'éviter toute exposition inutile de fichiers sensibles.

3.3 Correction de l'injection SQL dans le code PHP [/1]

Localisez dans le code PHP l'injection SQL exploitée dans la partie 4 et proposez une correction.

L'injection SQL est dans le fichier login.php, dans le bloc qui traite le POST. Elle se situe à l'endroit où la requête est construite en concaténant directement $username et $password dans la chaîne SQL. La correction consiste à remplacer cette concaténation par une requête préparée.

```
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $username = $_POST['username'];
    $password = md5($_POST['password']);

    try {
        $stmt = $mysqli->prepare("SELECT * FROM users WHERE username = ?
            AND password = ?");
        if (!$stmt) {
            throw new Exception("Error: " . $mysqli->error);
        }

        $stmt->bind_param("ss", $username, $password);
        $stmt->execute();
        $result = $stmt->get_result();
    } catch (Exception $e) {
        echo $e->getMessage();
        exit;
    }

    if ($result->num_rows != 0) {
        $_SESSION['loggedin'] = true;
        $_SESSION['username'] = $username;
        header("location: index.php");
```

```
    } else {
        $showError = true;
    }
}
```

3.4 Mécanisme de basculement des services [/2]

En vous basant sur le code source PHP et Javascript de l'application web, expliquez précisément le fonctionnement du mécanisme qui permet de basculer l'état des services.

Le tableau de bord lance automatiquement une mise à jour périodique via Javascript. La fonction updateStatus crée une requête HTTP GET vers le script status.php chaque seconde. La réponse est du JSON. Ce JSON contient l'état des services web, ftp et ssh, ainsi que le uptime. Le code Javascript parse la réponse puis met à jour les éléments de la page en ajustant les classes CSS des indicateurs. Les classes running, stopped et notfound déterminent la couleur affichée.

Le basculement de l'état se fait au clic sur un indicateur. Chaque bouton appelle toggleService avec un identifiant de service prédéfini dans le onclick, par exemple apache2, vsftpd ou ssh. toggleService envoie ensuite une requête HTTP GET vers toggleService.php en ajoutant le paramètre service dans l'URL.

Côté PHP, toggleService.php commence par vérifier que la session existe et que l'utilisateur est authentifié. Ensuite, une vérification supplémentaire impose que l'utilisateur soit admin. Si ce n'est pas admin, le script répond immédiatement avec un JSON qui indique un échec et un message.

Si l'utilisateur est admin, le script récupère le paramètre service depuis la requête. Il appelle checkServiceStatus depuis functions.php pour déterminer si le service est running ou non. Selon le statut, le script exécute une commande système via shell_ exec en envoyant soit start soit stop vers le programme /usr/bin/controller, en lui passant le nom du service. La sortie de cette commande est placée dans un objet JSON success true et renvoyée au navigateur.

Enfin, côté Javascript, la requête de basculement ne met pas directement à jour les pastilles. L'affichage est rafraîchi par les appels périodiques à status.php, ce qui reflète l'état réel après l'action.

4 Compromission d'un compte utilisateur [/4]

4.1 Récupération du hash de carol par injection SQL [/2]

En utilisant une injection SQL, récupérez le hash du mot de passe de carol.

Votre réponse ici.

4.2 Craquage du hash [/1]

Cassez le hash pour obtenir le mot de passe de carol. Indice : le mot de passe est composé uniquement de 6 chiffres.

Votre réponse ici.

4.3 Récupération de controller.c via SSH [/1]

Connectez-vous en SSH au compte de carol, et récupérez le fichier `controller.c`.

Votre réponse ici.

5 Élévation de privilèges [/6]

Raccourci : Le fichier `controller.c` est disponible sur Moodle si non obtenu à la partie précédente.

5.1 Shell www-data par injection de commande PHP [/2]

Utilisez une injection de commande PHP dans le mécanisme qui permet de basculer l'état des services pour obtenir un shell en tant que `www-data`.

Votre réponse ici.

5.2 Analyse de controller.c et vulnérabilité buffer overflow [/2]

Que fait le programme `controller.c` ? Identifiez une vulnérabilité causée par un débordement de tampon et expliquez comment la corriger.

Votre réponse ici.

5.3 Exploitation du buffer overflow pour devenir root [/2]

Exploitez le dépassement de tampon dans le programme `controller` et devenez root.

Votre réponse ici.

6 Accès physique = Game Over [3 points bonus]

Obtenez un shell root sur la machine virtuelle TP2.

Votre réponse ici.

Bibliographie

Indiquez toutes vos sources d'information (humaines ou documentaires).

Références

[1] nmap – Network Mapper. `https://nmap.org/`