

2023

Rapport Pentest DemoDay

Zoheir KICHOU

03/02/2023

# Introduction

# Contexte et objectifs du test d'intrusion :

* + Ce rapport de test d'intrusion a été réalisé pour JEDHA School, du 01 au 03 février 2023.
  + Le test a été mené par KICHOU Zoheir, un expert en cybersécurité.
  + L'adresse IP autorisée à tester était 172.31.35.242. Il a été demandé de ne pas toucher au port 9999 durant les tests.
  + Le type de test d'intrusion effectué était un test général, ce qui signifie que toutes les parties du système ont été testées pour déterminer leur vulnérabilité.
  + Le test d'intrusion a abouti à la découverte de 9 vulnérabilités.
  + Les résultats du test et les recommandations pour la correction de ces vulnérabilités sont détaillés dans les sections suivantes de ce rapport.

# Méthodologie utilisée

* + Reconnaissance : Collecte d'informations sur l’adresses IP, les systèmes d'exploitation, les applications, les ports ouverts, etc.
  + Scannage : Vérification de la présence de vulnérabilités en utilisant des outils automatisés pour scanner les ports, les services et les systèmes.
  + Exploitation : Tentative d'exploiter les vulnérabilités identifiées lors de la phase de reconnaissance et de scannage.
  + Vérification : Vérification de l'exploitation réussie pour déterminer si la vulnérabilité est effectivement présente.
  + Rapport : Documentation des vulnérabilités identifiées, des méthodes d'exploitation utilisées et des résultats obtenus.

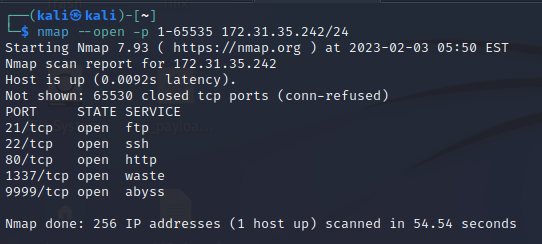
# Collecte d’informations active

Lors d'un scan de l'adresse 172,31,35,242, les résultats ont révélé que plusieurs ports étaient ouverts :

* Sur le port 21, un service FTP était en cours d'exécution.
* Le port 22, un service SSH était disponible.
* Le port 80 hébergeait un service HTTP.
* Le port 1337 était associé au service waste.
* Le port 9999, un service appelé abyss était en fonctionnement.

Il est important de noter que la présence de ports ouverts peut indiquer une vulnérabilité potentielle pour la sécurité de l'adresse 172,31,35,242.

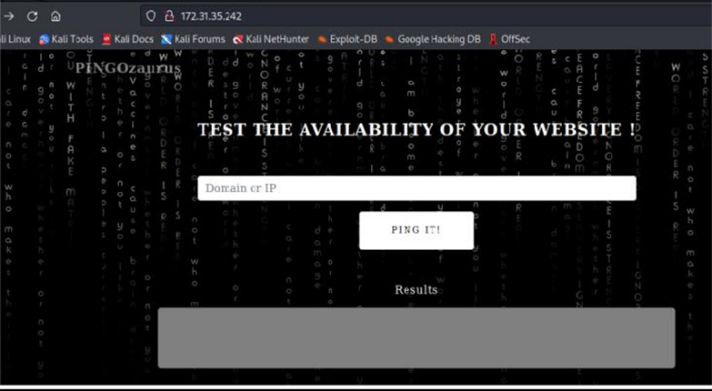
Il est donc crucial de prendre les mesures nécessaires pour s'assurer que ces services sont configurés de manière sécurisée pour minimiser les risques pour la sécurité.

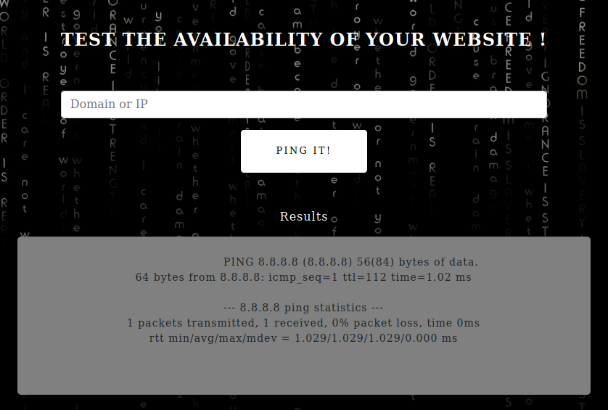


# Détails des vulnérabilités identifiées (niveau de gravité, description, preuve de concept, etc.)

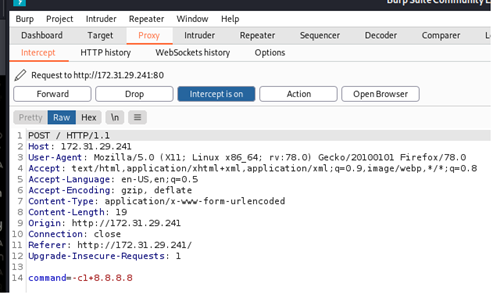
# Vulnérabilité 01 (l’user W) et Vulnérabilité 05 (l’user W vers J) :

On est allé sur le site web hébergé sous ce serveur 172.31.35.242, on a trouvé un site qui proposait de faire un ping d’une adresse web ou une IP.

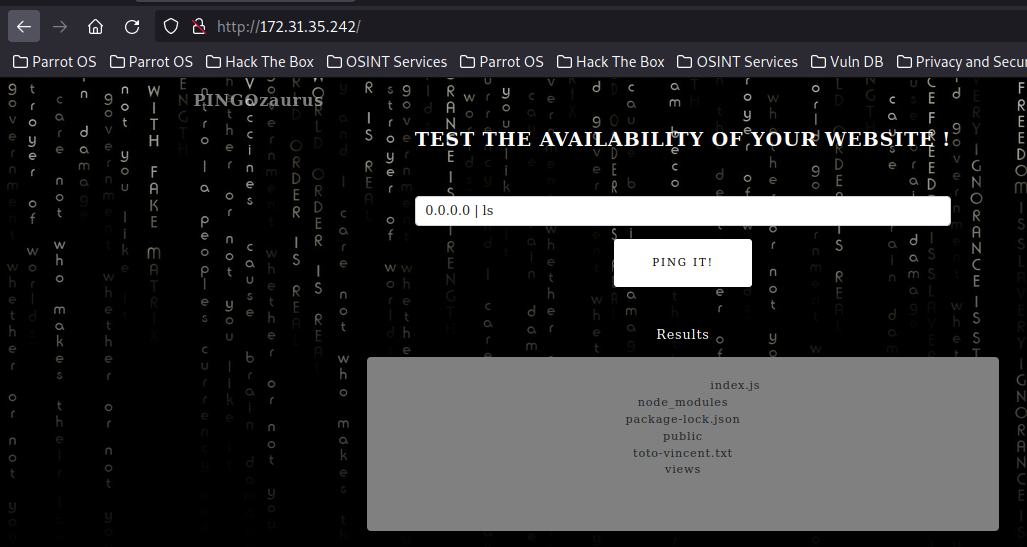




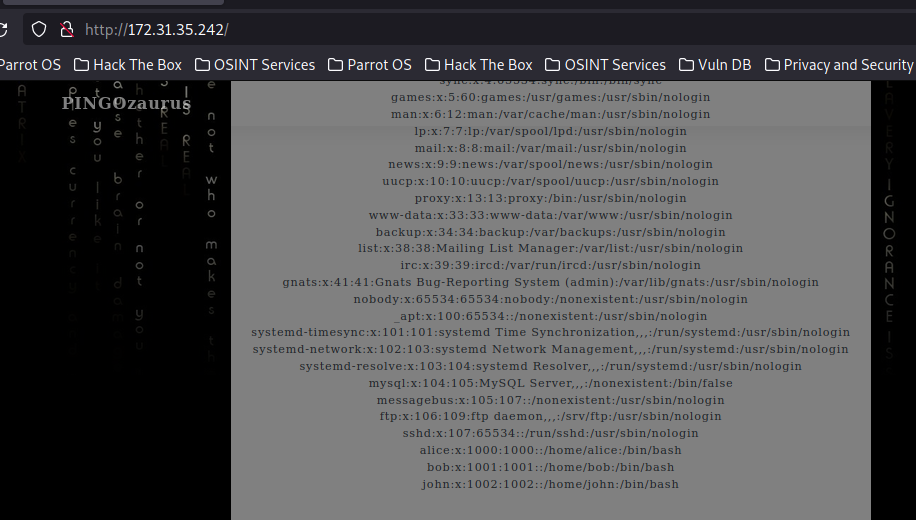
On a testé un ping classique, grâce à l’outil Burp, on remarque que la commande de la requête est simplement ce qui est rentré dans le formulaire.



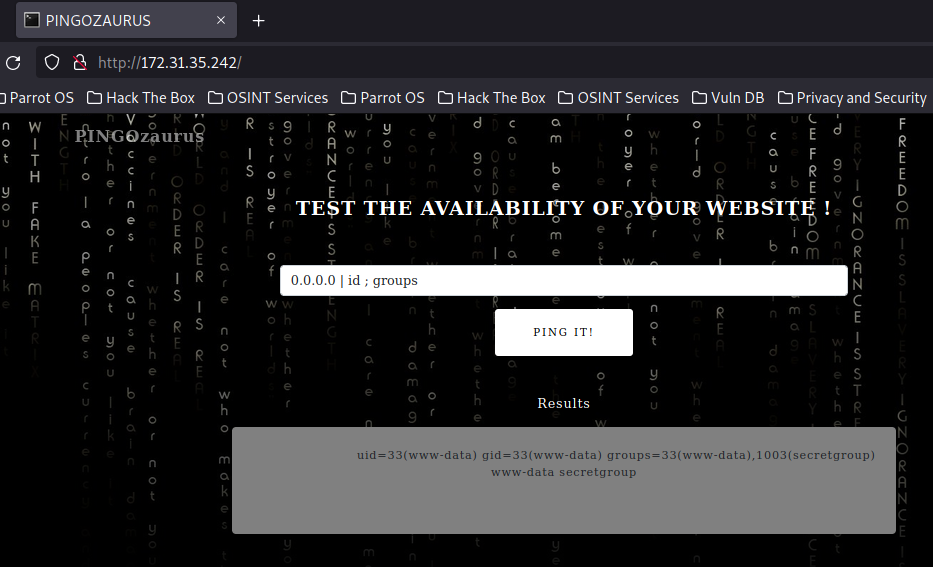
On a essayé de changer cette commande afin d’envoyer également au serveur une commande shell par exemple « **0.0.0.0 | ls** », on remarque que le serveur a bien exécuté notre commande.



On lance la commande suivante : **« 0.0.0.0 | cat /etc/passwd** » pour avoir la liste des users, le résultat nous a permis de récupérer tous les utilisateurs



Puis avec la commande : « **0.0.0.0 | id ; groups** », on a pu connaitre l’user [www.data](http://www.data) et qu’il appartient au groupe « secretgroup »



**Exploitation de la Vulnérabilité 05 (l’user W vers J)**

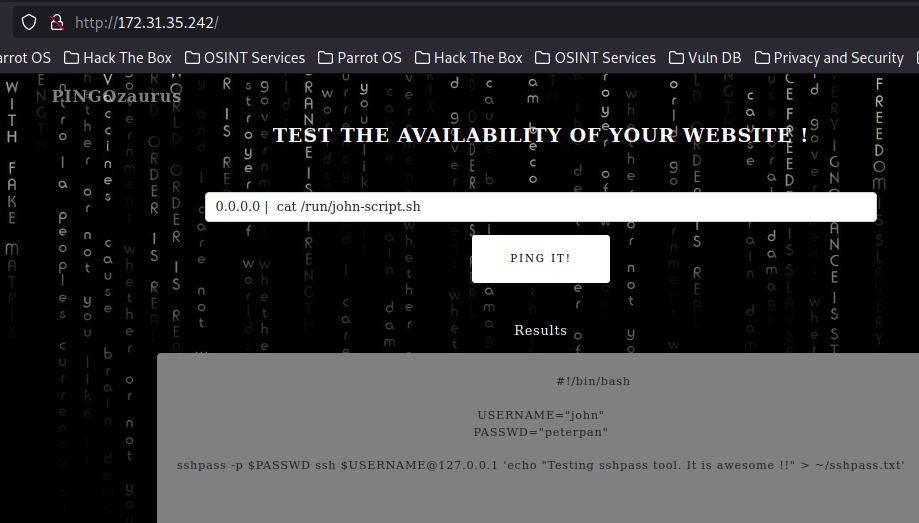
Grâce à **LinEnum**, à partir de l’utilisateur Alice, on a découvert le fichier suivant :

**-rw-r----- 1 john secretgroup 160 Oct 21 13:00 /run/john-script.sh**

Ce fichier appartenant au groupe **secretgroup**, on peut le lire via une injection à partir du site Web à l’aide de la commande suivante :

* + 1. **| cat /run/john-script.sh**

On vient de récupérer les credentials (USERNAME="john" et PASSWD="peterpan") de l’utilisateur John à partir du site Web.



# Vulnérabilité 02 (l’user J) : Port 22 ouvert SSH

PS : Grâce à la vulnérabilité n°1, on a trouvé les credentials (USERNAME="john" et PASSWD="peterpan") de l’utilisateur John, mais voyons si y’a pas un autre moyen de les avoir,

Pour cela, on va tenter un brute force avec l’outil « Hydra » via la commande suivante :

**hydra -l john -P /usr/share/wordlists/rockyou.txt 172.31.35.242 ssh**

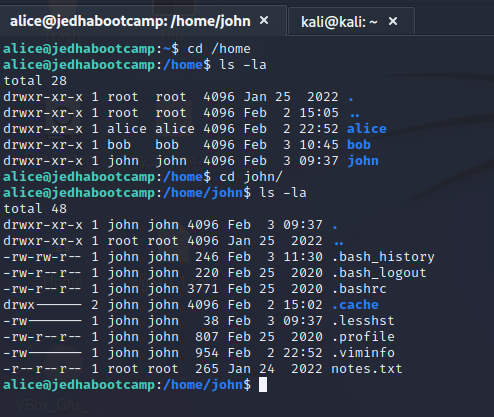
Notre brute-force a bien fonctionné, le mdp a été trouvé **« peterpan »**



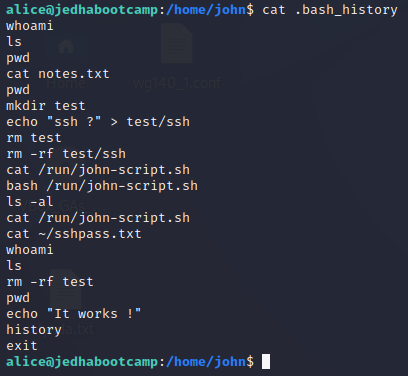
**Une autre façon de le faire :**

Grace à la vulnérabilité 03 (détaillée ci-dessous) un ID\_RSA a été récupéré dans le serveur FTP, en utilisant cet ID, on a pu se connecter sur l’user A « Alice »,

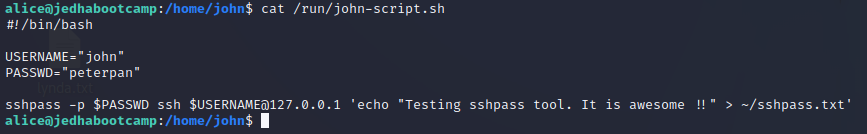
Puis passer sur l’user J (John),



Ce dernier n’avait pas supprimé l’historique de ses commandes BASH\_HISTORY,



Avec une simple commande CAT, on a eu accès à son historique ou son MDP été enregistré.



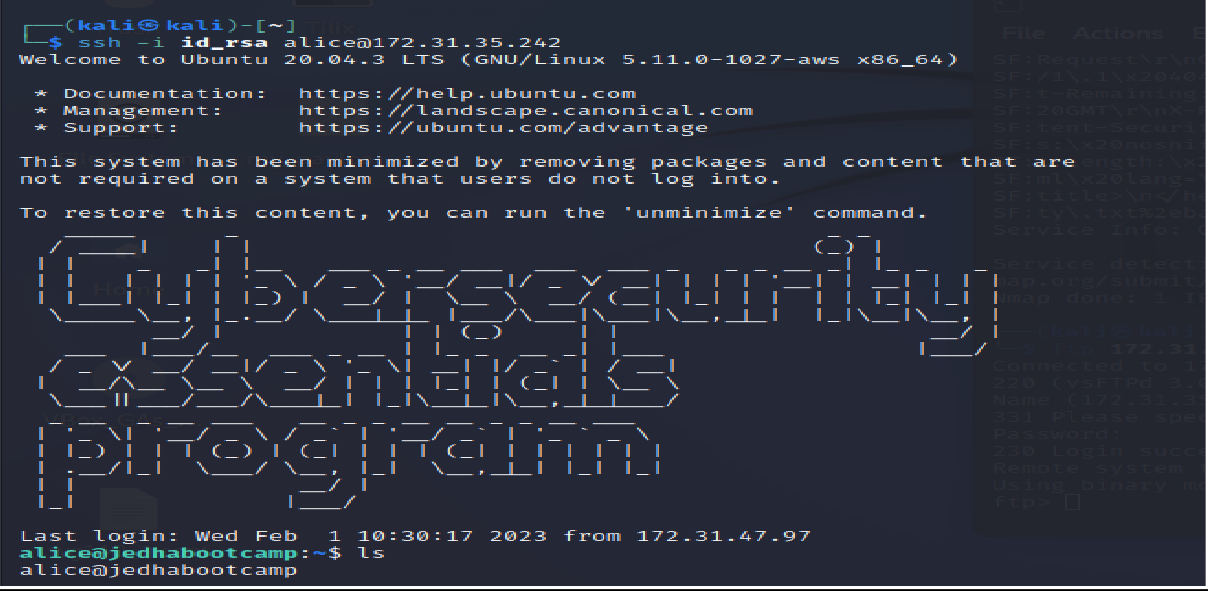
# Vulnérabilité 03 (l’user A) : port FTP Anonymous autorisé

Le port 21 FTP était aussi ouvert et il autorisé les connexions anonymes,

En passant par ce port, on a pu récupérer le ID\_RSA de l’user (ALICE),

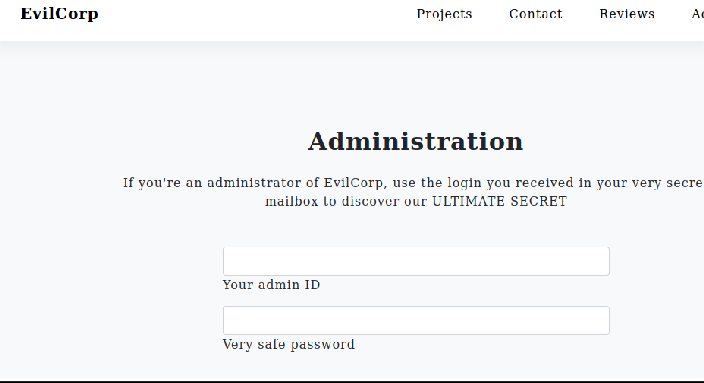


On l’a utilisé pour se connecter sur sa session avec la commande SSH -i ID\_RSA [alice@172.31.35.242](mailto:alice@172.31.35.242)

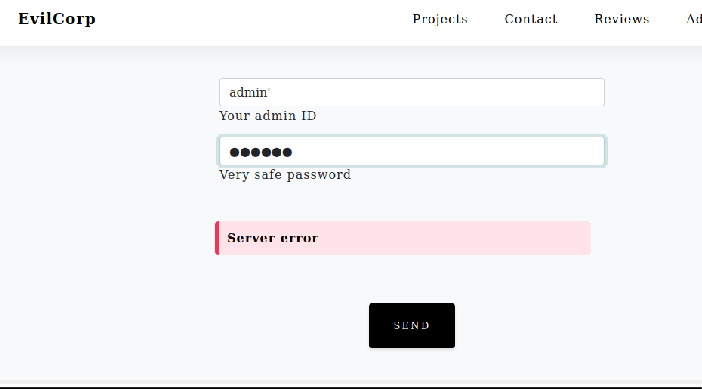


# Vulnérabilité 04 (l’user B) : Injection SQL

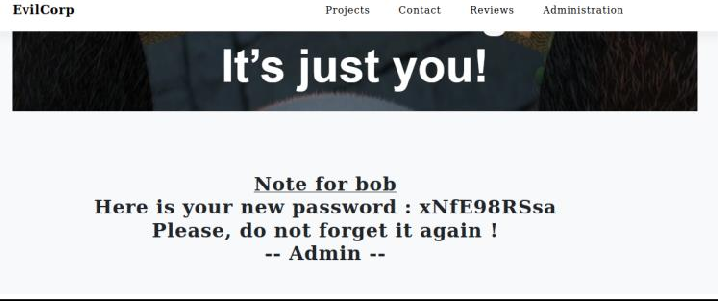
Une fois sur le site « EvilCorp », champs « Administration »



Un test d’injection a été effectué pour voir si le site-web filtre ou pas les injections, le test était positif pour nous, (erreur 500) a été détectée,



Grâce à une injection bien ciblée « **' OR 1=1# »**, le mdp de BOB a été révélé.



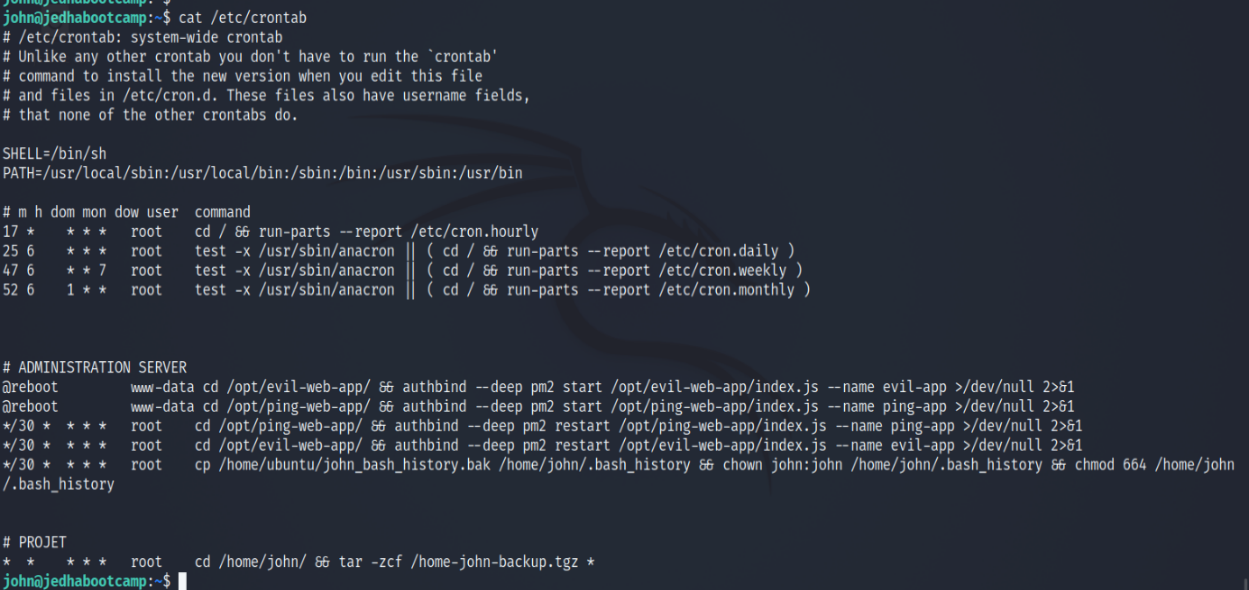
Nous tentons de nous connecter avec les credentials de Bob à l’aide de la commande suivante : **sshpass -p xNfE98RSsa ssh** [**bob@172.31.35.242**](mailto:bob@172.31.35.242)



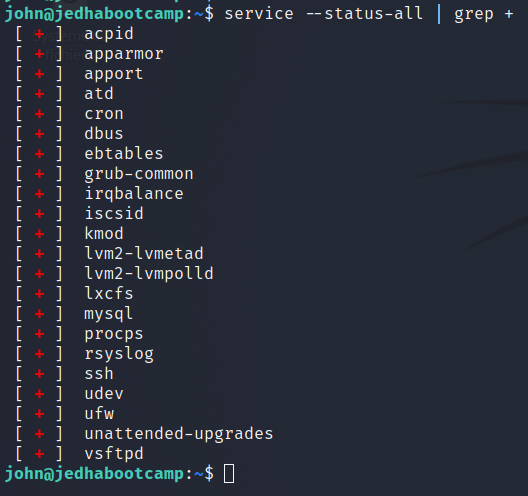
# Vulnérabilité 06 (l’user J vers R) :

En éditant le fichier **« /etc/crontab »,** nous découvrons la ligne suivante permettant d’effectuer une sauvegarde régulière du /home de John :

**\* \* \* \* \* root cd /home/john/ && tar -zcf /home-john-backup.tgz \***

****

De plus, à l’aide de la commande « **service --status-all | grep +** » nous voyons bien que le service CRON est bien actif.

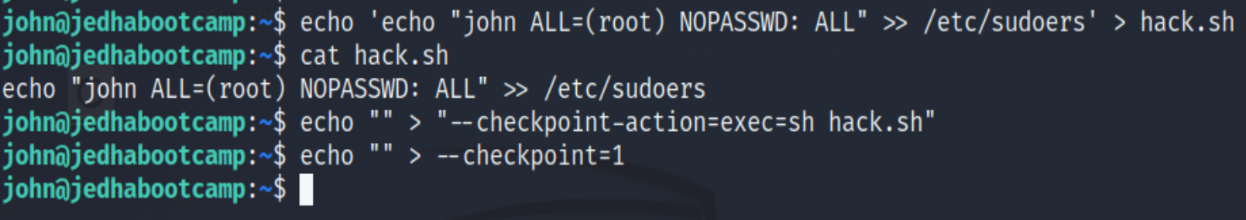


Nous devons exécuter les commandes suivantes afin de donner les droits « **root »** à l’utilisateur John :

**echo 'echo "john ALL=(root) NOPASSWD: ALL" >> /etc/sudoers' > hack.sh**

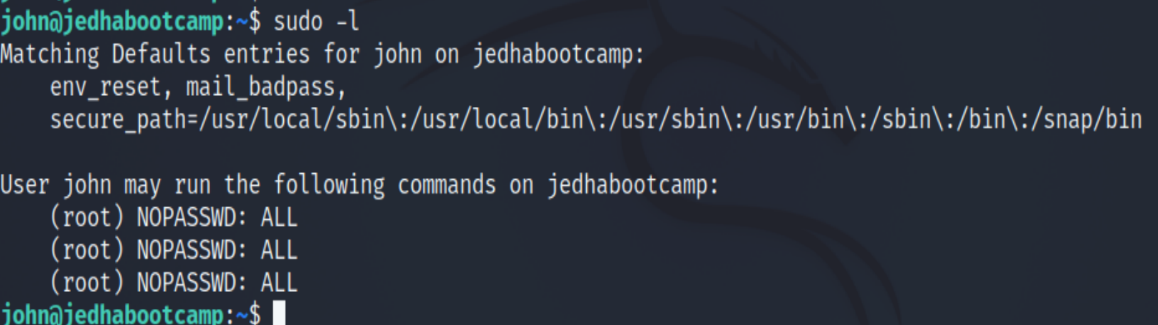
**echo "" > "--checkpoint-action=exec=sh hack.sh"**

**echo "" > --checkpoint=1**

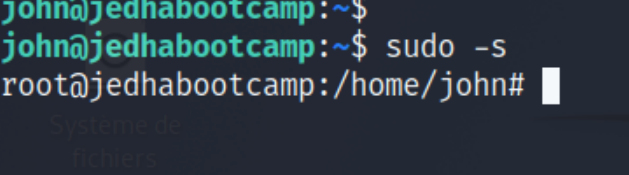


Une fois que la tâche CRON est de nouveau lancée en root, nous pouvons exécuter la commande

« **sudo -l »** à partir de l’utilisateur John pour vérifier que ce dernier fait bien dorénavant parti des « **sudoers** »



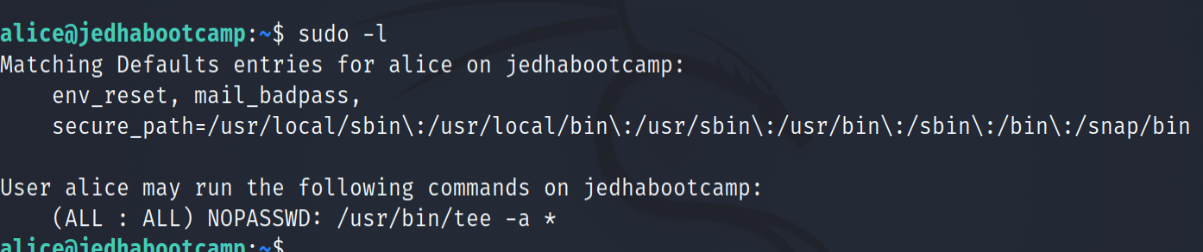
Dorénavant, l’utilisateur John appartenant bien **sudoers**, nous pouvons effectuer la commande **sudo -s** afin de lancer un shell root.



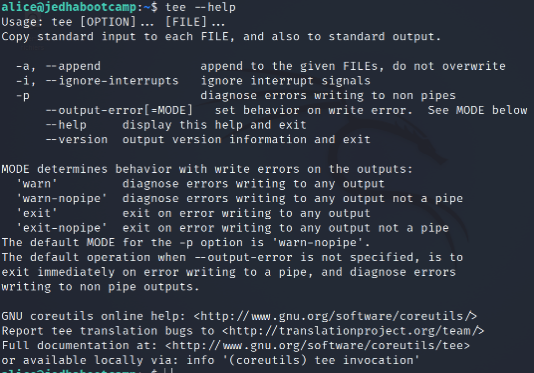
# Vulnérabilité 07 (l’user A vers R) :

Une fois connecté avec l’utilisateur Alice, nous allons essayer de déterminer si cette dernière possède des droits **sudoers** afin d’élever nos privilèges et d’éventuellement passer Root à l’aide de la commande suivante : **sudo –l**

Effectivement, nous avons visé juste car Alice peut lancer la commande **tee –a** en sudo sans demande de mot de passe.



A l’aide de la commande « **tee –help** », nous remarquons que l’option **–a** correspond à la concaténation de fichiers.



Par conséquent, nous allons ajouter un compte **root** au sein du fichier « **/etc/passwd »**

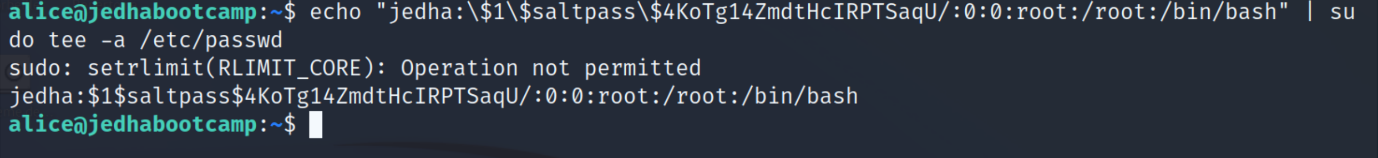
Pour cela, nous allons créer un compte « **jedha »** avec le mot passe « **jedha »** qui est hashé à l’aide de la commande suivante : » **openssl passwd -1 -salt saltpass jedha »** et qui donne le mot de passe hashé suivant : **$1$saltpass$4KoTg14ZmdtHcIRPTSaqU/**

Donc au final, nous allons ajouter la ligne suivante au sein du fichier « **/etc/passwd »**:

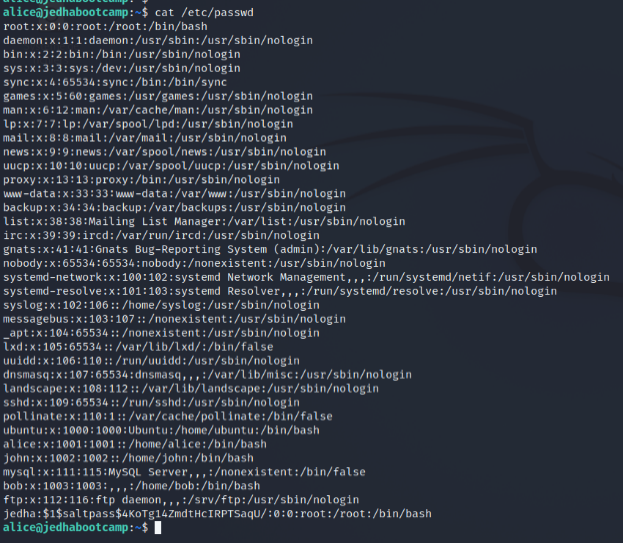
**« jedha: $1$saltpass$4KoTg14ZmdtHcIRPTSaqU/:0:0:root:/root:/bin/bash »** à l’aide de la commande tee.

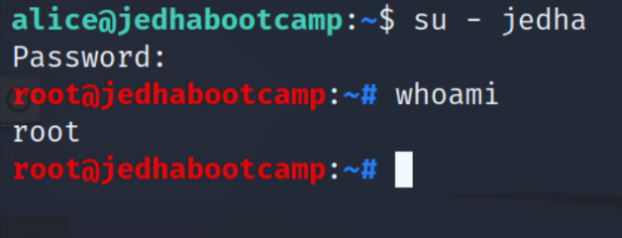
Pour ajouter, notre compte **root** à partir d’Alice au sein du fichier « **/etc/passwd** », nous lançons la commande suivante :

**« echo "jedha:\$1\$saltpass\$4KoTg14ZmdtHcIRPTSaqU/:0:0:root:/root:/bin/bash" | sudo tee -a /etc/passwd »** attention à bien « back slasher » les $.



A l’aide d’un **cat** sur le fichier « **/etc/passwd »**, nous voyons que notre nouveau compte **root** a bien été créé et que nous pouvons nous y connecter en effectuant un « **su – jedha »** avec le mot de passe « **jedha »**

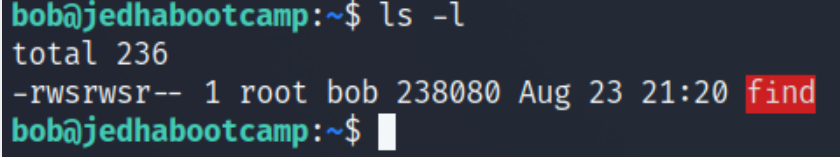
****

****

# Vulnérabilité 08 (l’user B vers R) :

Une fois connecté avec l’utilisateur Bob, nous remarquons la présence d’un binaire **find** possédant les droits suivants :

**-rwsrwsr-- 1 root bob 238080 Aug 23 21:20 find**

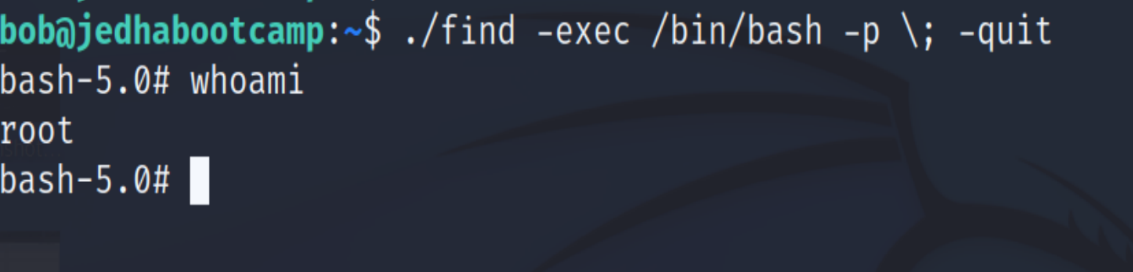


Par conséquent, d’après les droits ci-dessus (SUID et SGID), nous remarquons que les utilisateurs du groupe **bob** peuvent exécuter ce binaire **find** avec des droits **root.**

Or étant donné que nous pouvons lancer des commandes à l’aide de l’option **-exec**, nous allons tenter de lancer un **bash** en tant que **root** à l’aide de la commande suivante :

**./find -exec /bin/bash -p \; -quit**

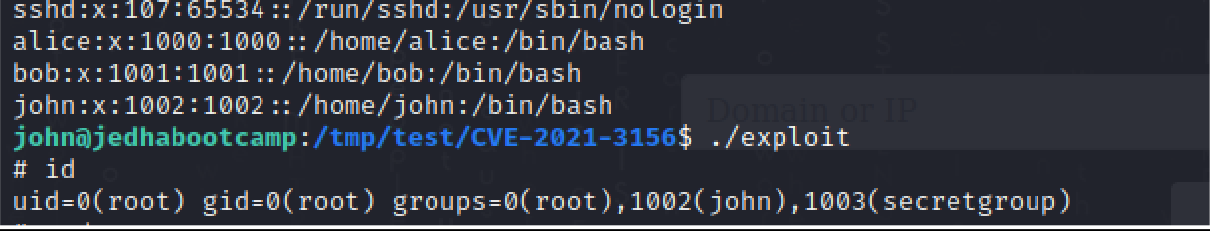
Nous remarquons que cela fonctionne parfaitement et que nous sommes dorénavant **root**.



# Vulnérabilité 09 (les users J, A et B vers R) : CVE-2021-3156

On a utilisé l’exploit CVE-2021-3156 (lien dispo dans les annexes), grâce à cet exploit, on a pu élever les privilèges des 3 users à Root.

On a créé un dossier qui se nommait TEST sous le chemin suivant john/tmp/, puis on a lancé l’exploit, ce dernier nous a permis de passer à root



# Recommandations pour la correction des vulnérabilités identifiées

|  |  |  |
| --- | --- | --- |
| N° | Vulnérabilité | Contre-mesures |
| 1 | Validation insuffisante des entrées utilisateur dans un formulaire web | Toujours valider et filtrer les entrées utilisateur dans les formulaires web pour éviter les attaques par injection de code ou autres techniques malveillantes. |
| 2 | Mot de passe vulnérable à une attaque par force brute | Utiliser une politique de mot de passe forte, comprenant des caractères alphanumériques et spéciaux et une longueur suffisante pour rendre l'attaque par force brute plus difficile. |
| 3 | Compte anonyme sur un serveur FTP et stockage de fichiers sensibles en clair | Ne pas autoriser l'utilisation de comptes anonymes sur un serveur FTP et chiffrer les fichiers sensibles avant de les stocker sur le serveur. |
| 4 | Vulnérabilité d'injection SQL et stockage de mots de passe en clair | Valider toutes les entrées utilisateur dans les formulaires d'authentification et ne jamais stocker les mots de passe en clair. |
| 5 | Stockage de  « credentials » d'utilisateur en clair dans un fichier système | Ne jamais stocker les « credentials » d'utilisateur en clair dans un fichier système, utiliser plutôt des méthodes de stockage sécurisées telles que des fichiers de configuration chiffrés. |
| 6 | Autorisation de la concaténation de fichiers avec les droits « root » pour un utilisateur sans privilèges élevés | Ne jamais autoriser la concaténation de fichiers avec les droits « root » pour les utilisateurs qui n'en ont pas besoin et limiter les droits d'accès aux fichiers sensibles aux personnes autorisées. |
| 7 | Utilisation d'une tâche CRON pour l'archivage régulier du dossier /home d'un utilisateur | Être très vigilant quant aux tâches CRON qui s'exécutent, car elles sont lancées en tant que root, et utiliser des outils de surveillance pour détecter toute activité suspecte. |
| 8 | Utilisation de SUID et de SGID sur des binaires appartenant à l'utilisateur root | Ne jamais utiliser SUID et SGID sur des binaires appartenant à root, sauf si cela est strictement nécessaire, et limiter l'accès aux fichiers et aux répertoires sensibles aux personnes qui en ont besoin. |
| 9 | Utilisation d'une version vulnérable de « Sudo » | Faire une veille technologique pour être au courant des nouvelles failles de sécurité, mettre à jour régulièrement le système pour disposer des dernières versions des binaires et configurer correctement les politiques de sécurité pour minimiser les risques d'attaques. |

# Annexes

Liste des outils et techniques utilisées pour le test d'intrusion :

Nmap

Msfconsole

[GitHub - blasty/CVE-2021-3156](https://github.com/blasty/CVE-2021-3156)