RAPPORT PENTEST

TP3

08 NOVEMBRE 2024

BUT R&T SOPHIA ANTIPOLIS
MELLANO LANCELOT

Table des matières

Introduction :

Dans le cadre de notre formation en cybersécurité, nous avons participé à un exercice de test d'intrusion encadré visant le réseau cible 172.17.0.0/24. Cet exercice, autorisé et supervisé par nos professeurs Thomas PREVOST et Yohan BERTRAND, avait pour objectif de découvrir et d'exploiter des vulnérabilités spécifiques sur une machine située à l'adresse 172.17.0.2. Grâce à un accès sécurisé via OpenVPN, nous avons pu établir une communication avec la cible hébergée sur un serveur de l'IUT.

Les étapes de notre attaque comprenaient la découverte initiale du réseau, l'identification des vulnérabilités, puis l'exécution d'exploits, avec une tentative progressive d'élévation de privilèges pour atteindre des niveaux d'accès avancés au système cible. Les outils utilisés pour ce pentest incluaient Nessus, SQLmap, Nmap, atftp, et bkcrack, qui nous ont permis de mener différentes phases de notre attaque.

L'objectif principal de l'exercice consistait à localiser et extraire trois fichiers de validation : deux "intermediate flags" marquant des niveaux d'accès intermédiaires, ainsi qu'un "root flag" symbolisant le niveau d'accès le plus avancé.

Ce rapport présente en détail chaque étape de l'attaque en commençant par la phase de reconnaissance, les vulnérabilités identifiées, l'exploitation de ces vulnérabilités ainsi que les correctives à apporter pour sécuriser les failles de sécurités, dans le but de renforcer notre compréhension des menaces et des pratiques de sécurité adaptées.

Disclaimer :

Ce rapport présente les failles de sécurité découvertes dans le cadre d'un test d'intrusion réalisé sur une durée limitée de trois heures. Il est important de noter que d'autres vulnérabilités pourraient exister dans l'infrastructure ciblée, mais elles n'ont pu être détectées dans le temps imparti ou en raison de limitations techniques et méthodologiques propres à cet exercice.

Déroulé de l'attaque :

Reconnaissance :

Pour identifier la machine cible, nous avons commencé par vérifier la connectivité avec une commande ping vers l'adresse IP 172.17.0.2, confirmant ainsi que la machine cible était en ligne et accessible sur le réseau.

```
root@rtnnnpxx:~# ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=63 time=1.92 ms
64 bytes from 172.17.0.2: icmp_seq=2 ttl=63 time=0.972 ms
64 bytes from 172.17.0.2: icmp_seq=3 ttl=63 time=1.49 ms
```

Scan de vulnérabilité :

Dans le cadre d'un test d'intrusion, la phase de scan de vulnérabilités est essentielle après la reconnaissance initiale. Elle consiste à analyser en profondeur le système cible pour identifier les failles de sécurité exploitables.

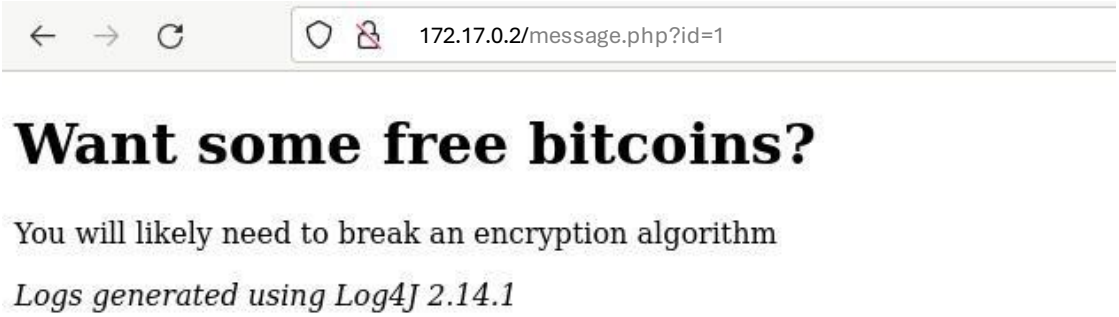
Commençons avec l'outil Nmap, reconnu pour sa puissance en matière de découverte réseau et d'analyse de ports. En exécutant la commande « nmap 172.17.0.2 -sT», nous avons pu obtenir une liste des ports ouverts sur la machine cible. Le scan a révélé que les ports et les services suivants étaient ouverts :

```
Nmap scan report for 172.17.0.2
Host is up (0.00044s latency).
Not shown: 995 closed tcp ports (reset)
PORT     STATE SERVICE
80/tcp   open  http
139/tcp  open  netbios-ssn
443/tcp  open  https
445/tcp  open  microsoft-ds
8009/tcp open  ajp13

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
root@rtnnnpxx:~#
```

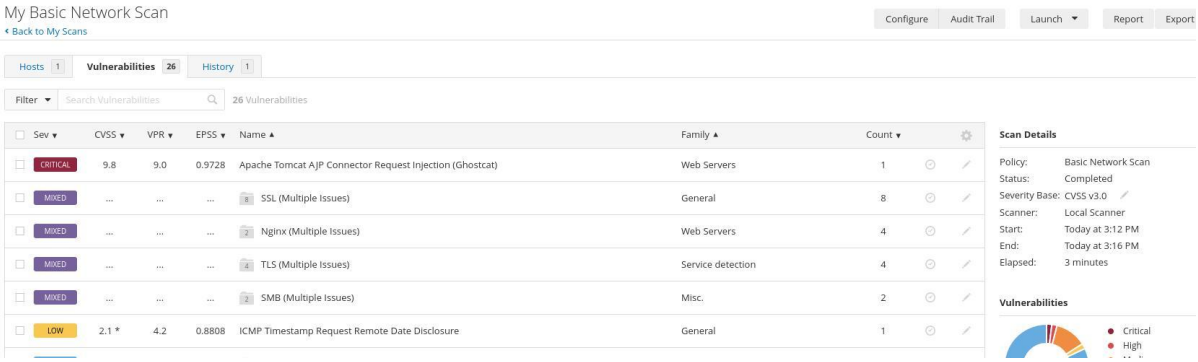
Ces résultats nous donnent une première idée des services en cours d'exécution sur la machine et des éventuelles vulnérabilités exploitables. On remarque par exemple que

le port http est ouvert ce qui nous informe qu'on peut surement interagir avec le site web hébergé par la machine :



On remarque qu'une page existe, à première vue elle n'est pas très intéressante mais on remarque dans l'url qu'il y a une variable qui est passé donc par l'intermédiaire de la méthode GET. Ce qui peut potentiellement mener à une vulnérabilité.

De plus nous utilisons Nessus qui est un outil de scan de vulnérabilités utilisé pour identifier les failles de sécurité sur des systèmes et des réseaux. Il est capable d'analyser des adresses IP, des ports ouverts et des configurations de services pour y détecter des failles connues.



Dans notre cas, le scan Nessus sur l'adresse 172.17.0.2 a révélé la présence d'une faille critique dans le service Apache Tomcat, connue sous le nom de *Apache Tomcat AJP Connector Request Injection*. Cette vulnérabilité permet à un attaquant d'injecter des requêtes malveillantes dans le service AJP (Apache JServ Protocol), qui est un protocole utilisé pour la communication entre Tomcat et d'autres serveurs.

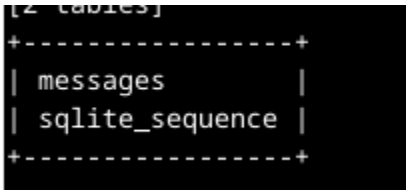
Exploitation :

Test d'injection SQL :

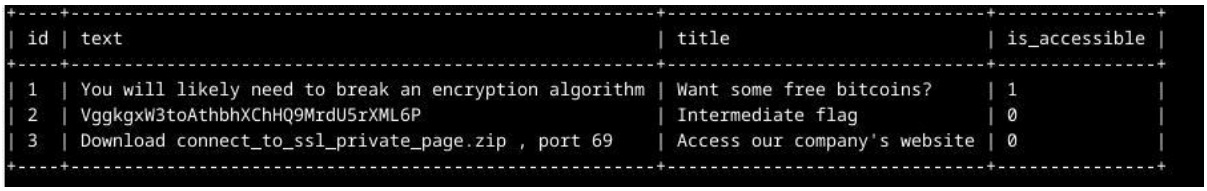
Pour donner suite au scan de vulnérabilité, nous allons tenter de voir si le site web héberger par notre cible possède des failles en tentant des injections SQL.

Pour cela nous utiliserons l'outil SQLmap. C'est un outil automatisé qui permet de détecter et d'exploiter des injections SQL sur les applications web. Dans notre cas, nous avons ciblé l'URL « http://172.17.0.2/message.php?id=1 » pour tester la présence d'une injection SQL.

En commençant par l'affichage des tables avec la commande « sqlmap -u "http://172.17.0.2/message.php?id=1" --tables » :



SQLmap a pu identifier deux tables dans la base de données : messages et sqlite_sequence. Cette première étape nous permet de repérer les structures de données disponibles, on va d'ailleurs exploiter cela en faisant un dump (copie complète de la table) de la table messages grâce à la commande « sqlmap -u "http://172.17.0.2/message.php?id=1" --dump -T messages » :



Le résultat révèle plusieurs lignes de données, incluant un « **intermediate flag »** dans le champ text d'id 2, et une autre ligne qui nous indique de télécharger un fichier connect_to_ssl_private_page.zip à télécharger sur le port 69.

Test du protocole TFTP :

Il faut savoir que le port 69, correspond au protocole tftp qui est un protocole de partage de fichier qui utilise UDP, donc c'est pour ça qu'on n'a pas vu le port 69 ouvert lors de notre premier nmap car nous avons ajouté l'option -sT pour avoir que les ports TCP, car si on affiche tous les ports(TCP et UDP), nous en auront beaucoup trop donc on filtre directement avec -sT. On fait alors un "nmap -sU 69 172.17.0.2 -p69", -sU permet d'indiquer qu'on veut scanner les ports utilisant UDP et le –p69 précise le port 69. On remarque donc que le port 69 est ouvert :

```
root@rtnnnpxx:~# nmap -sU 172.17.0.4 -p69
Starting Nmap 7.93 ( https://nmap.org ) at 2024-11-06 15:39 CET
Nmap scan report for 172.17.0.4
Host is up (0.00097s latency).

PORT   STATE         SERVICE
69/udp open|filtered tftp

Nmap done: 1 IP address (1 host up) scanned in 0.33 seconds
root@rtnnnpxx:~#
```

Nous essayons donc de nous connecter au serveur TFTP :

```
root@rtnnnpxx:~# tftp
(to) 172.17.0.2
tftp> ls
?Invalid command
tftp> get connect_to_ssl_private_page.zip
Transfer timed out.
```

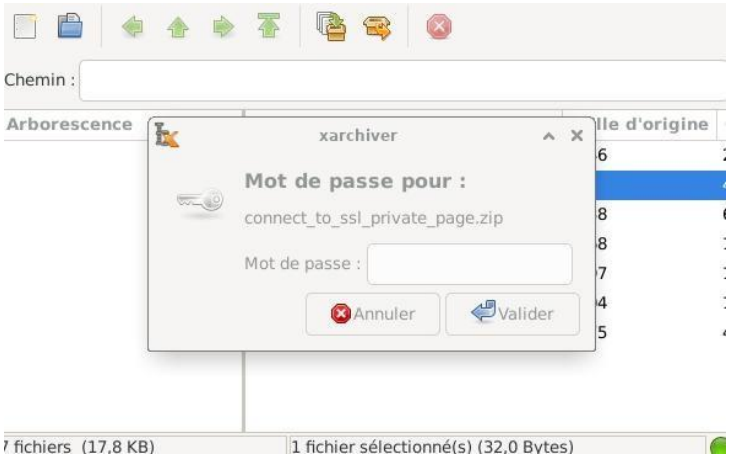
Or nous n'arrivons pas à récupérer le fichier, on remarque que nous avons un Transfer timed out.

Heureusement il n'existe pas que l'utilitaire tftp, on va donc essayer avec autre chose comme atftp:

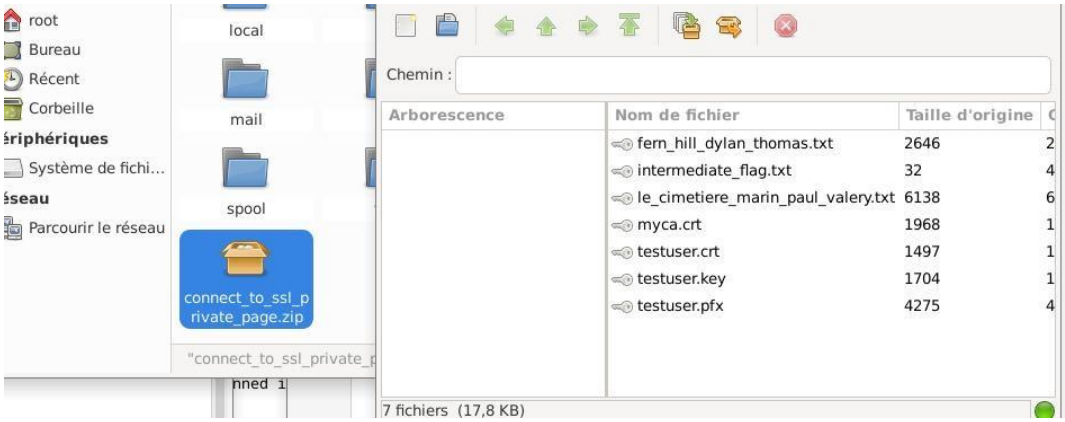
```
root@rtnnnpxx:/var# atftp
tftp> connect 172.17.0.4
tftp>
tftp>
tftp>
tftp>
tftp>
tftp> get connect_to_ssl_private_page.zip
tftp> exit
tftp: bad command name.
tftp> ^Z
[2]+  Stoppé                 atftp
root@rtnnnpxx:/var# ls
backups  connect_to_ssl_private_page.zip  local  log   opt  spool  www
cache    lib                              lock   mail  run  tmp
root@rtnnnpxx:/var#
```

Nous récupérons donc le fichier connect_to_ssl_private_page.zip

Or ce fichier nécessite un mot de passe pour extraire le contenue.



On peut juste remarquer qu'il comporte différents fichiers donc un intermediate_flag.txt :



Exploitation du fichier .zip :

Pour avoir plus d'informations sur les fichiers, nous pouvons afficher des informations techniques détaillées sur chaque fichier du dossier connect_to_ssl_private_page.zip sans les extraire en utilisant la commande "7z l -slt connect_to_ssl_private_page.zip" :

On peut donc remarquer la méthode utilisée pour le cryptage, le CRC, etc.

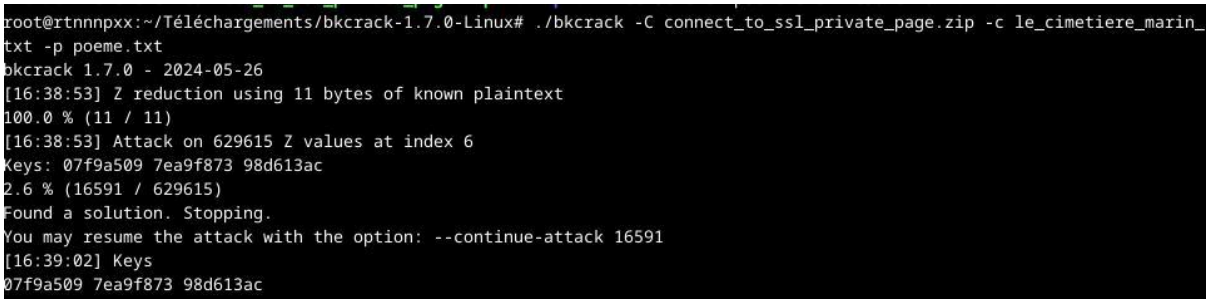
L'objectif est de trouver une manière d'obtenir ou de contourner le mot de passe pour avoir accès aux fichiers. En effectuant quelques recherches sur internet, nous sommes tombés sur l'utilitaire bkcrack qui est un outil utilisé pour craquer les mots de passe d'archives ZIP chiffrées avec un certain type de chiffrement dont ZipCrypto.

Pour ce faire on doit utiliser une commande qui nécessite :

- Un répertoire .zip
- Un fichier du répertoire qui est crypté
- Une partie du fichier crypté en claire

Or nous n'avons pas accès au contenu des fichiers du répertoire, on remarque qu'il y a un fichier le_cimetière_marin_paul_valery.txt qui est un poème. On peut alors tenter de récupérer sur internet le début du poème que l'on copie dans un fichier texte.

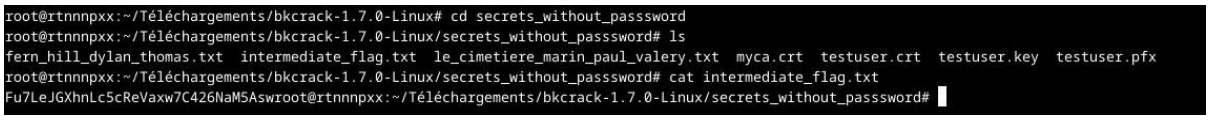
On entre ensuite la commande de bkcrack “. /bkcrack -C connect_to_ssl_private.zip -c le_cimetiere_marin_paul_valery.txt -p poeme.txt”



Nous parvenons à obtenir la clé de déchiffrage, bkcrack possède d’autre fonctionnalités, nous allons en utiliser une qui permet de créer une nouvelle archive .zip basée sur connect_to_ssl_private.zip, mais sans protection par mot de passe.

Grâce à la commande ”./bkcrack -C connect_to_ssl_private.zip -k 09f9a509 7ea9f873 98d613ac -D secrets_without_password.zip”:

On remarque qu’on obtient bien dans notre répertoire courant le fichier secrets_with_new_password.zip. On peut alors extraire les fichiers puis afficher ”l’intermediate_flag.txt” :



Fu7LeJGXhnLc5cReVaxw7C426NaM5Asw

Correctives à mettre en place :

Faille SQL :

Vulnérabilité :

La vulnérabilité d'injection SQL est une faille critique classée dans le Top 10 OWASP et est souvent exploitée par des attaquants pour accéder à des données sensibles. Elle se produit lorsque des paramètres utilisateurs (comme id=1 dans une requête GET) sont directement intégrés dans des requêtes SQL sans vérification ni validation. Cette faille est classée de sévérité élevée car elle peut permettre à un attaquant de lire, modifier ou supprimer des données dans la base de données cible.

Correctives :

- **Utilisation de requêtes préparées pour les paramètres GET :** L'application devrait utiliser des requêtes préparées pour gérer les paramètres GET. Cela empêche un attaquant de modifier le contenu du paramètre id pour injecter des commandes SQL.
- **Validation de la variable id :** Il faut s'assurer que la valeur du paramètre id respecte le type attendu. Dans ce cas, si id doit être un entier, vérifiez qu'il est bien numérique avant de l'inclure dans une requête SQL.

Faille TFTP :

Vulnérabilité :

De plus, l'utilisation du protocole TFTP (Trivial File Transfer Protocol) pour héberger des fichiers critiques représente une faille de sécurité majeure. TFTP est un protocole non sécurisé, ne supportant ni l'authentification ni le chiffrement, et donc inadéquat pour transférer des fichiers sensibles. La criticité de cette faille est élevée, car un attaquant peut facilement accéder au fichier si l'emplacement est découvert.

Correctives :

- **Retirer/Dissimuler les instructions d'accès** : Retirez la ligne indiquant le téléchargement de fichiers sensibles via TFTP (download connect_to_ssl_private_page.zip), car elle fournit un indice direct aux attaquants.
- **Remplacer le protocole TFTP :** TFTP est un protocole peu sécurisé qui ne prend pas en charge l'authentification ni le chiffrement, rendant les fichiers accessibles à quiconque connaît leur emplacement. Il est recommandé de remplacer TFTP par un protocole sécurisé comme SFTP car ce protocole offre des fonctionnalités d'authentification et de chiffrement.

- **Implémentation de l'authentification :** Pour limiter l'accès aux fichiers, implémentez une authentification stricte pour que seuls les utilisateurs autorisés puissent les télécharger.

Faille du fichier .zip :

Vulnérabilité :

La méthode de chiffrement utilisée pour sécuriser le fichier .zip est ZipCrypto, qui est un chiffrement faible et obsolète. Des outils comme bkcrack peuvent facilement le déchiffrer, ce qui expose les fichiers à des attaques. Cette vulnérabilité est classée de sévérité élevée en raison de la facilité avec laquelle elle peut être exploitée pour contourner la sécurité.

Correctives :

- **Utiliser un chiffrement robuste** : Remplacez la méthode de chiffrement ZipCrypto par AES-256 pour chiffrer les fichiers .zip. AES-256 est considéré comme une norme de chiffrement sécurisé et offre une meilleure protection contre les attaques par force brute et les outils de déchiffrement.
- **Utiliser des mots de passe forts** : Assurez-vous que les fichiers chiffrés sont protégés par des mots de passe complexes et longs, rendant plus difficile le cassage du mot de passe, même avec un chiffrement fort.