GTFOBins

GTFOBins rappresenta uno strumento molto efficace per chi si occupa di sicurezza informatica, in particolare per l'escalation di privilegi o il bypass delle restrizioni di sicurezza su sistemi Unix o Linux. La sua peculiarità risiede nel fatto che sfrutta binari legittimi già presenti sui sistemi target, rendendo queste tecniche meno sospette rispetto a metodi più tradizionali. Il concetto chiave è che, se configurati in modo insicuro, questi programmi possono essere utilizzati per ottenere un accesso con privilegi superiori o eseguire comandi arbitrari.

Per esempio, se ci si trova su un sistema con privilegi limitati, il primo passo logico sarebbe verificare quali comandi è possibile eseguire con permessi elevati. Un buon punto di partenza è eseguire il comando ‘sudo -l’, che elenca i comandi che è possibile eseguire con sudo senza dover inserire la password. Supponiamo che tra questi ci siano binari come ‘find’, ‘awk’ o ‘less’, strumenti comunemente presenti su molti sistemi. Ognuno di questi binari può essere sfruttato per ottenere un accesso privilegiato.

Un esempio classico riguarda ‘find’, un comando solitamente utilizzato per cercare file all'interno del filesystem. Tuttavia, con l'opzione giusta, ‘find’ può eseguire comandi arbitrari. Se il binario può essere eseguito con privilegi elevati, il seguente comando permette di ottenere una shell root: ‘sudo find . -exec /bin/bash \’

In questo modo, si ottiene l'accesso alla shell con privilegi di root, una escalation di privilegi attraverso un binario legittimo.

Un altro binario interessante è ‘awk’, normalmente utilizzato per l'analisi e l'elaborazione di testo. Anche in questo caso, è possibile sfruttarlo per eseguire comandi di sistema. Con ‘awk’, è sufficiente eseguire: ‘sudo awk 'BEGIN {system("/bin/bash")}'’

Anche qui, si ottiene una shell privilegiata in pochi passaggi, utilizzando un comando ordinario.

Perfino strumenti apparentemente innocui come ‘less’, utilizzato per visualizzare file, possono essere sfruttati per ottenere privilegi superiori. Se si dispone dei permessi necessari per eseguire ‘less’, si può lanciare una shell all'interno del comando stesso, ad esempio eseguendo: ‘sudo less /etc/passwd’ e poi ‘! bash’.

Questo apre una shell direttamente all'interno di less, fornendo accesso completo al sistema.

Quindi, il processo per sfruttare GTFOBins è relativamente semplice: prima si utilizza sudo -l per elencare i comandi che possono essere eseguiti con sudo. Successivamente, si consulta il sito ufficiale di GTFOBins per verificare quali di questi comandi possono essere usati. Ogni binario presente su GTFOBins è accompagnato da una serie di esempi pratici che spiegano come sfruttarlo per ottenere una shell o eseguire comandi con privilegi elevati.

Ad esempio, possiamo vedere che se si dispone del comando tar, uno dei più comuni, si può utilizzare la seguente sequenza per ottenere una shell privilegiata: ‘sudo tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/bash’.

L'idea fondamentale è quella di sfruttare binari legittimi per ottenere accesso non autorizzato al sistema, senza dover ricorrere a tecniche più invasive o visibili. In un contesto di penetration testing, la capacità di individuare e utilizzare tali binari può rappresentare un vantaggio decisivo, poiché permette di aggirare le difese di un sistema sfruttando componenti che l'amministratore potrebbe non percepire come rischiosi.

Conoscere quali binari cercare e come sfruttarli è essenziale per il successo in questo tipo di scenari.

PwnKit (CVE-2021-4034)

PwnKit (CVE-2021-4034) rappresenta una vulnerabilità critica scoperta nel sistema polkit, un componente presente su molte distribuzioni Linux che gestisce le autorizzazioni a livello di sistema. La vulnerabilità permette di eseguire comandi come root senza autenticazione, sfruttando il comando 'pkexec', una funzionalità di polkit che consente di eseguire programmi con privilegi elevati. Questa falla è estremamente pericolosa perché è presente in molte versioni di polkit distribuite con diversi sistemi operativi Linux e permette di ottenere privilegi di root con un attacco relativamente semplice.

Per comprendere meglio questa vulnerabilità e sfruttarla in un contesto di penetration testing, partiamo dall'identificare se un sistema è vulnerabile. Prima di tutto, puoi verificare la presenza di polkit sul sistema in questione usando comandi come:

- ✓ 'dpkg -l | grep polkit # su sistemi Debian/Ubuntu'
- ✓ 'rpm -qa | grep polkit # su sistemi RedHat/CentOS'

Se il sistema ha una versione di polkit vulnerabile (prima delle patch rilasciate all'inizio del 2022), potresti essere in grado di sfruttare la vulnerabilità per ottenere una shell privilegiata.

Il problema principale in PwnKit risiede, però, nel modo in cui pkexec gestisce gli argomenti passati al comando. In pratica, se viene eseguito senza argomenti, pkexec non controlla correttamente l'input, permettendo all'attaccante di manipolare il comando e far eseguire codice arbitrario con privilegi di root.

Per sfruttare questa vulnerabilità, gli attaccanti possono creare uno script exploit che sfrutta questa mancanza di controllo. Uno degli exploit più comuni si trova già pubblicamente disponibile su GitHub, pronto per essere utilizzato su sistemi vulnerabili. Di seguito un esempio semplificato dei passaggi che si potrebbero seguire per sfruttare questa vulnerabilità:

- ✓ Individua la vulnerabilità: come detto, verifica se polkit è installato e controlla la versione. Se è una versione vulnerabile, sei a buon punto.
- ✓ Scarica l'exploit: puoi trovare exploit già pronti per CVE-2021-4034 su repository pubblici come GitHub. Uno degli exploit più noti per PwnKit è uno script scritto in C che compila un eseguibile che sfrutta la falla in pkexec.
- ✓ Compila l'exploit: se stai usando uno degli exploit pubblici, puoi compilarlo direttamente sulla macchina vulnerabile. Per farlo, scarica il codice exploit e compila il file con un comando come: 'gcc -o exploit pwnkit_exploit.c'. Questo produrrà un eseguibile chiamato exploit che può essere eseguito sul sistema vulnerabile.
- ✓ Esegui l'exploit: una volta compilato l'exploit, basta eseguirlo per ottenere una shell privilegiata. Esegui: './exploit'.

Se il sistema è vulnerabile e l'exploit funziona correttamente, otterrai una shell root, concedendoti pieno accesso al sistema.

In conclusione, la vulnerabilità PwnKit è particolarmente pericolosa perché può essere sfruttata senza necessità di autenticazione e non richiede interazione da parte dell'utente. Questo rende gli attacchi estremamente efficaci e difficili da rilevare, specialmente in ambienti in cui i sistemi non vengono aggiornati frequentemente. Molte distribuzioni Linux hanno fornito patch per risolvere questa vulnerabilità, ma resta comunque importante sapere come funziona nel contesto di penetration testing e come individuarla in ambienti non sicuri.

Per proteggere i sistemi da questa vulnerabilità, è fondamentale aggiornare il pacchetto polkit all'ultima versione disponibile, che include la patch per CVE-2021-4034, eseguendo un semplice aggiornamento di sistema.

Per chi si occupa di sicurezza, è essenziale comprendere come funziona il bug e come sfruttarlo in contesti di test o dimostrazioni, ma soprattutto è cruciale sapere come rilevare e patchare questa vulnerabilità per proteggere i sistemi dagli attacchi.

Knockd

knockd è un demone che implementa una tecnica chiamata port knocking, una misura di sicurezza usata per nascondere i servizi di rete dietro firewall, rivelandoli solo a chi invia una sequenza di pacchetti TCP o UDP specifica. È come una serratura segreta: solo chi conosce la sequenza corretta può accedere a un servizio nascosto, come ad esempio SSH.

La logica alla base di knockd è che un servizio, come una porta SSH, resta invisibile alle scansioni esterne fino a quando non riceve una 'knock sequence', una serie predefinita di tentativi di connessione a diverse porte.

Immaginiamo di voler proteggere un servizio SSH che, per ragioni di sicurezza, vogliamo tenere nascosto finché non è effettivamente necessario accedervi. Con knockd, si puòconfigurare il sistema in modo che il servizio SSH rimanga chiuso e invisibile finché qualcuno non invia la sequenza corretta di pacchetti alle porte specificate. Una volta ricevuta la sequenza, knockd apre temporaneamente la porta SSH, permettendo l'accesso solo a chi conosce la sequenza.

Per iniziare, si deve installare il pacchetto knockd sul sistema con il comando: 'sudo apt install knockd'.

Una volta installato, si deve modificare il file di configurazione di knockd, che di solito si trova in /etc/knockd.conf. Un esempio di configurazione è questo che riporto nell'immagine sottostante, in cui nascondiamo il servizio SSH sulla porta 22, e lo facciamo aprire solo dopo una sequenza di knock su tre porte (7000, 8000, 9000).

La sequenza di knock per aprire la porta SSH è 7000, 8000, 9000. Il comando iptables permette l'accesso all'IP del client che ha inviato la sequenza.

La sequenza inversa 9000, 8000, 7000 chiude nuovamente la porta.

seq_timeout indica il tempo massimo (in secondi) entro cui la sequenza deve essere completata per essere considerata valida.

```
[options]
        UseSyslog

[openSSH]
        sequence    = 7000,8000,9000
        seq_timeout = 5
        command     = /sbin/iptables -A INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn

[closeSSH]
        sequence    = 9000,8000,7000
        seq_timeout = 5
        command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn
```

Una volta configurato il file, possiamo avviare knockd con: ‘sudo systemctl start knockd’.

Ora, la porta SSH resterà chiusa finché non verrà "bussato" alle porte nella sequenza corretta.

Si può accedere al servizio SSH anche da un altro sistema, importante è inviare la sequenza corretta di pacchetti. Questo può essere fatto utilizzando il comando knock, disponibile in molti sistemi Linux: ‘knock indirizzo_IP_target 7000 8000 9000’.

Se la sequenza è corretta, knockd aprirà temporaneamente la porta 22 (SSH) sul sistema target, permettendoti di connetterti. Una volta terminato, si chiude la porta inviando la sequenza inversa: ‘knock indirizzo_IP_target 9000 8000 7000’.

Il port knocking è una misura di sicurezza che aggiunge un ulteriore livello di protezione ai servizi di rete, ma non è infallibile. Un attaccante che conosce la sequenza potrebbe comunque accedere al servizio. Tuttavia, knockd offre il vantaggio di rendere più difficile per gli attaccanti identificare i servizi esposti tramite normali scansioni di rete.

Una tecnica più sicura potrebbe essere l'uso di Single Packet Authorization (SPA), che è un’evoluzione del port knocking in cui un singolo pacchetto crittografato contiene le informazioni necessarie per autenticare l’accesso. Tuttavia, knockd è ancora molto utile in molti scenari, specialmente quando è combinato con altre misure di sicurezza come firewall e sistemi di monitoraggio, quindi utilizzato come parte di una strategia di sicurezza più ampia.