

48730/32548, Cyber Security

Lab1, Part 2

Password Cracking using John the Ripper and SQL Injection

The Tasks included in Lab 1, Part 2 require VMware Workstation or VMware Fusion. For this session, only Cybersec-Server image is required.

Lab 1, Part 2 should be submitted along with Lab 1, Part 1 in Week 3 (refer to assignment section of the course on Canvas for the Due date). The assignment document should contain only questions from the lab manual, your answers and screenshots to support your answers.

Password Cracking using John the Ripper

Introduction

John the Ripper is an Open-Source password security auditing and password recovery tool available for many operating systems. It is one of the most popular password testing and breaking programs. JtR autodetects the encryption on the hashed data and compares it against a large plain-text file that contains popular passwords, hashing each password, and then stopping it when it finds a match. JtR also includes its own wordlists of common passwords for 20+ languages. These wordlists provide JtR with thousands of possible passwords from which it can generate the corresponding hash values to make a high-value guess of the target password. Since most people choose easy-to-remember passwords, JtR is often highly effective even with its out-of-the-box wordlists of passwords.

We already have John the Ripper installed in our server, if you wish to download and install JtR, you can go to Open wall's Website [here](#) or from the Official John the Ripper Repo [here](#).

 **Required resources: Cybersec-Server.**

Task 4: Use John the Ripper to crack password

Step 1: Run John the Ripper.

- a. To see a list of commands in JtR:

```

cybersec-server@ubuntu:~$ john
John the Ripper password cracker, version 1.8.0
Copyright (c) 1996-2013 by Solar Designer
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single                "single crack" mode
--wordlist=FILE --stdin wordlist mode, read words from FILE or stdin
--rules                enable word mangling rules for wordlist mode
--incremental[=MODE]   "incremental" mode [using section MODE]
--external=MODE        external mode or word filter
--stdout[=LENGTH]     just output candidate passwords [cut at LENGTH]
--restore[=NAME]       restore an interrupted session [called NAME]
--session=NAME         give a new session the NAME
--status[=NAME]        print status of a session [called NAME]
--make-charset=FILE    make a charset, FILE will be overwritten
--show                show cracked passwords
--test[=TIME]          run tests and benchmarks for TIME seconds each
--users=[-]LOGIN|UID[,..] [do not] load this (these) user(s) only
--groups=[-]GID[,..]   load users [not] of this (these) group(s) only
--shells=[-]SHELL[,..] load users with[out] this (these) shell(s) only
--salts=[-]N           load salts with[out] at least N passwords only
--save-memory=LEVEL    enable memory saving, at LEVEL 1..3
--node=MIN[-MAX]/TOTAL this node's number range out of TOTAL count
--fork=N              fork N processes

```

We can see the version we have is 1.8.0 and the usage of JtR. John the Ripper works in 3 distinct modes to crack passwords:

- Single mode: is the fastest and best mode if you have a full password file to crack.
- Wordlist mode: compares the hash to a known list of potential password matches.
- Incremental mode: is brute force mode that tries every possible character combination to find a result.

- b. Change to **Documents** directory, you will find "mypasswd" file, check the content of the mypasswd.

```

cybersec-server@ubuntu:~/Documents$ cat mypasswd
cybersec-server:$6$.HzX0ral$x.ie52E8I4ZC1/FTbiqsEWLG/hH7eri1lzDr7c/XBbniBCixgHej
1000:1000:CyberSec:/home/cybersec-server:/bin/bash
Alice:$6$pHP7TsIr$Y5LNFbieD4kdaGhp9jcDo532St./a6.bLlgaJtq7588HBWDDyevB9/oksQR6oA
:/home/Alice:
Eric:$6$qYmN1.xs$L1nwW.S46Yx4ciCyamvSW6vlf1QmPnGbI2QIeCRLPs7bluJpjCpbDMezsBw0Yoy
/home/Eric:
Bob:$6$ZqiJ2Ai/$aHiRq4wX0EZQvBnzeA8w6VhQDVUKNSGmM7IeyloQ3/z34QzRE4/GmNAI6UP4I.R.
home/Bob:
Eve:$6$ydZ/4DCp$3cWJ/Kq1r0aqpj5AFZ/SKc0N/9aPkgRvH2m2DRwTdgaaJ9.oEr0T09XecyKzceIq
home/Eve:

```

This file includes user's name and their encrypted password

Step 2: Recover Passwords.

- a. Type the following command in terminal: `cybersec-server@ubuntu:~$ john --show mypasswd`

```
cybersec-server@ubuntu:~/Documents$ john --show mypasswd
0 password hashes cracked, 5 left
cybersec-server@ubuntu:~/Documents$
```

As shown above, there are no cracked passwords at this point.

- b. At the command prompt, enter `john mypasswd` to use single crack mode. You can see the password for cybersec-server is cracked instantly, user Alice and Bob's password are cracked in seconds. `cybersec-server@ubuntu:~$ john mypasswd`

```
cybersec-server@ubuntu:~/Documents$ john mypasswd
Loaded 5 password hashes with 5 different salts (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
cybersec          (cybersec-server)
password          (Alice)
123456            (Bob)
```

Password cracking is CPU-intensive and a lengthy process, so the time it takes will depend on your system and the strength of the password. It can take days. If the password is not cracked for days with a powerful CPU, it is an incredibly good password.

Use Ctrl-C to abort the program after couple of minutes, then use `john --show` to check the same file again, we find there are 3 passwords cracked, 2 left.

```
cybersec-server@ubuntu:~/Documents$ john --show mypasswd
cybersec-server:cybersec:1000:1000:CyberSec:/home/cybersec-server:/bin/bash
Alice:password:1002:1002:./home/Alice:
Bob:123456:1004:1004:./home/Bob:

3 password hashes cracked, 2 left
```

- c. JtR uses a predefined dictionary called `password.lst` (in `/usr/share/john` directory) with a standard set of predefined "rules" for handling the dictionary and retrieves all password hashes of both md5crypt and crypt type, we have copied this file to our working directory and modified it.

Now let us use wordlist mode to crack the remaining passwords. This time user Eve's password is cracked instantly, that is because the password's file include this password.

```
cybersec-server@ubuntu:~/Documents$ john mypasswd --wordlist="password.lst"
Loaded 5 password hashes with 5 different salts (crypt, generic crypt(3) [?/64])
Remaining 2 password hashes with 2 different salts
Press 'q' or Ctrl-C to abort, almost any other key for status
Pa$$w0rd          (Eve)
lg 0:00:00:11 34% 0.08703g/s 91.90p/s 100.2c/s 100.2C/s seattle..789456
Use the "--show" option to display all of the cracked passwords reliably
Session aborted
```

The results below display the passwords for each account.

```
cybersec-server@ubuntu:~/Documents$ john --show mypasswd
cybersec-server:cybersec:1000:1000:CyberSec:/home/cybersec-server:/bin/bash
Alice:password:1002:1002:./home/Alice:
Bob:123456:1004:1004:./home/Bob:
Eve:Pa$$w0rd:1005:1005:./home/Eve:
4 password hashes cracked, 1 left
```

Mangling is a pre-processor in JtR that optimizes the wordlist to make the cracking process faster. Use the `--rules` parameter to set the mangling rules if you wish.

```
cybersec-server@ubuntu:~/Documents$ john --wordlist=password.lst --rules mypasswd
```

Challenge:

- 1) Can you find out the password for user Eric? (Screenshot required)

- 2) What did you learn from the password cracking process? How to create a secure password?

SQL Injection

SQL injection is one of the most common vulnerabilities in web applications today. It is one of the web hacking techniques that is immensely popular and dangerous because successful SQL injection could allow hackers to compromise your servers, networks, personal computers, and confidential data. According to The Open Web Application Security Project Report released in 2017, SQL Injection is amongst the number 1 risks out of top 10 security risks.

What is SQL Injection?

SQL injection is an attack injection technique that exploits vulnerability in SQL query via user's input data from client to the database layer of an application. This vulnerability exists in custom Web application that lacks proper input validation, fails to use parameterized SQL statements, and/or creates dynamic SQL with user-supplied data.

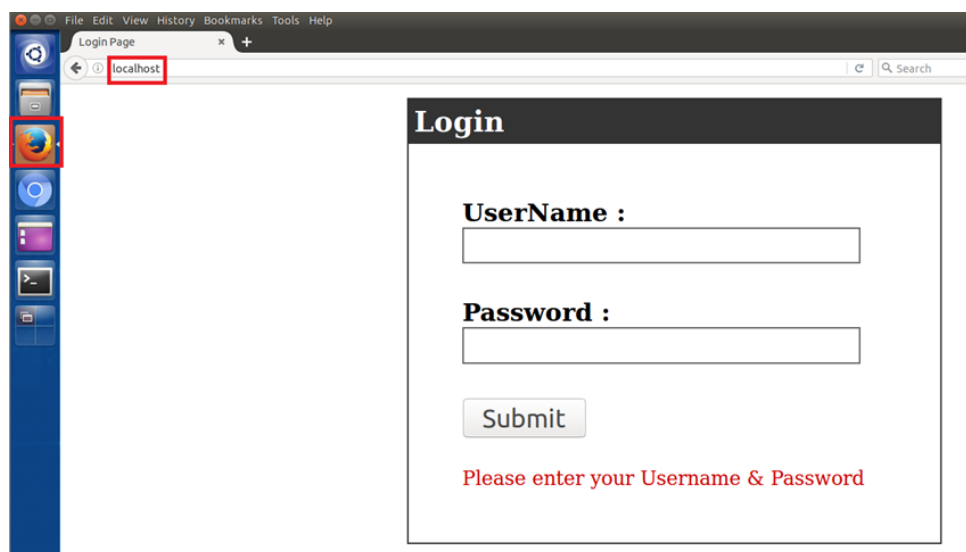
It is occurred when user input is incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. Normally, attacker will test SQL injection by typing malformed SQL commands into front-end Web application input boxes that are tied to database accounts to trick the database into offering more access to information than the developer intended.

A successful SQL injection exploit can read sensitive data from the database, modify database data, execute administration operations on the database, and recover the content of a given file present on the database file system and in some cases issue commands to the operating system. This attack allows attackers to spoof identity, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, destroy the data and become administrators of the database server.


Task5: SQL Injection

 **Resource Required: Cybersec-Server**

You need to use the browser and open localhost to perform this lab.



Login bypass is without a doubt one of the most popular SQL injection techniques. This lab will give explanations and a little deep understanding with some new flavours of bypasses.



Login

UserName :

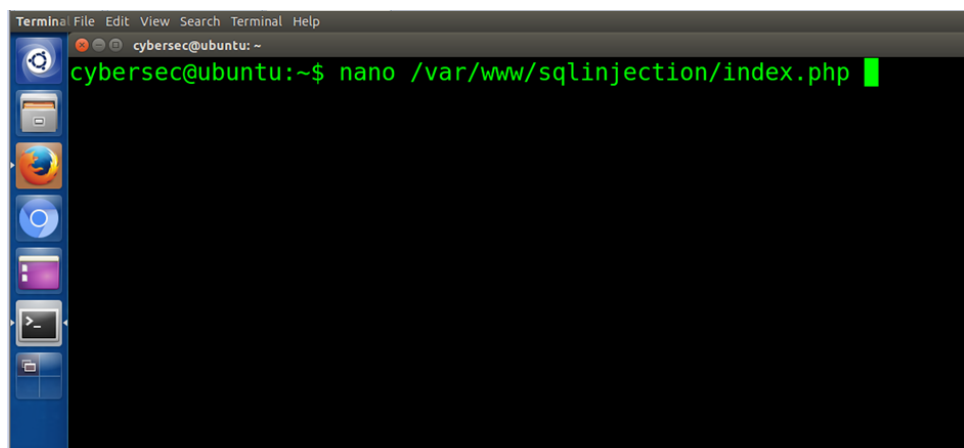
Password :

Submit

Error : Please check the Username and Password : You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '''' and password='' at line 1

In the image above, `'` was written in username and password was kept empty. When the Submit button was clicked, it showed an error message. From the error message, we can determine that MySQL is being used as the database.

Let us go to the Linux Terminal Window and enter the following command to open the source file:



```
cybersec@ubuntu: ~$ nano /var/www/sqlinjection/index.php
```

```

File Edit View Search Terminal Help
GNU nano 2.2.6 File: /var/www/sqlinjection/index.php

<?php
$host="localhost"; // Host name
$username="root"; // Mysql username
$password="root"; // Mysql password
$db_name="mydatabase"; // Database name
$tbl_name="members"; // Table name

// Connect to server and select database.
mysql_connect("$host", "$username", "$password")or die("cannot connect");
mysql_select_db("$db_name")or die("cannot select DB");

// Define $myusername and $mypassword
$myusername=$_POST['myusername'];
$mypassword=$_POST['mypassword'];

//Create the SQL query to check the username and password
$sql="SELECT username,password FROM $tbl_name WHERE username='$myusername' and password='$mypassword'";
$result=mysql_query($sql);
$rows=mysql_fetch_array($result);

// If result matched $myusername and $mypassword, $row must be true
if($rows){
    header("location: welcome.php");
}
elseif($myusername == '' and $mypassword == '')
{
    // If both fields are empty, show an error message
    header("location: login1.php");
}
else{
    // If the username or password is incorrect, show an error message
    header("location: login1.php");
}

```

Note: Please work through the information in the following link to understand how SQL query works:

- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- <https://sechow.com/bricks/docs/login-1.html>

Refer to the links provided above to perform the following tasks:

1. Login with Username as '123456789' and the Password as a SQL command to gain unauthorized access.
2. Login with both Username and Password as SQL commands.
3. Find table details containing all the Usernames and Passwords through SQL injection.
4. Login into a specific user account by extracting the username and password from the table.

Provide screenshots to support your answers, also mention the SQL commands used to gain unauthorized access.

Based on your Observations from the above task, suggest defence in your lab submission document.