Lab Sheet 3

Security

In our third lab, we will use hydra, a tool in Kali Linux, to perform a Brute Force attack on a Web based login. Hydra is the fastest network logon cracker which supports numerous attack protocols. It is very fast and flexible, and new modules are easy to add. This tool makes it possible for researchers and security consultants to show how easy it would be to gain unauthorized access to a system remotely.

1 Task 1: Brute Force Web-based Login with hydra

We will use http to run a brute force attack on a vulnerability testing site.

1.1 Get to know your tool

Type `hydra -h` (`h` for help) in your Kali terminal and find out what *hydra* is all about and how it might work.

1.2 Intro

One of hydra's brute force attacking services is used on Web-based logins, such as social media login forms, user banking login forms, your router's Web-based login, etc. That's `http[s]-get|post-form` which will handle this request.

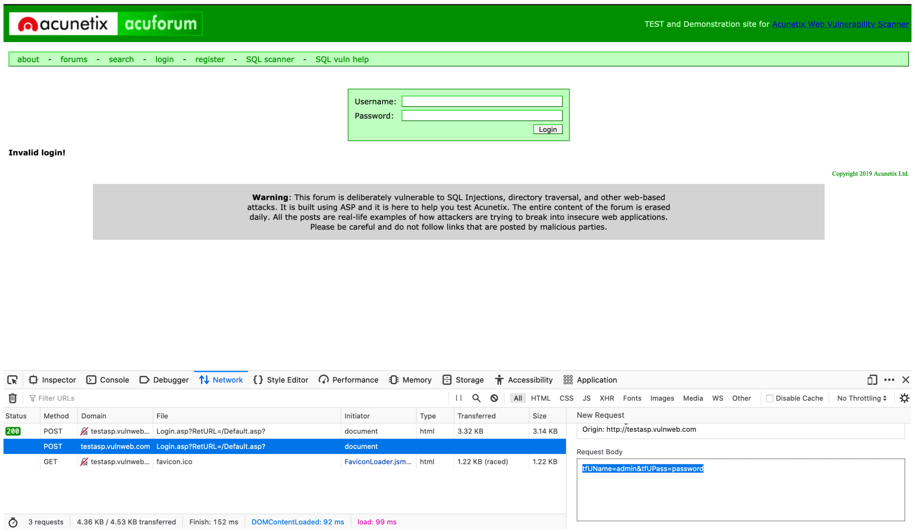
Before we fire up hydra, we should know some of the arguments:

- **Target:** http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault%2Easp%3F
- **Login username:** admin (if you're not sure, bruteforce this)
- **Password list:** "The location of the dictionary file list containing possible passwords."
- **Form parameters:** "Here we will be using iceweasel or firefox network developer toolbar."
- **Service module:** http-post-form

1.3 Obtaining post parametres

Visit http://testasp.vulnweb.com/Login.asp?RetURL=%2FDefault%2Easp%3F and open the *Network* tool in the *Web Developer* menu.

To obtain the post-form parameters, type anything in the *username* and/or *password* form. You will notice a new **POST** method on the network developer tab. Click on that line and on the *Headers* tab click *Edit and Resend* on the right side. From the *Request Body* textbox, copy the last line, e.g. `tfUName=admin&tfUPass=password`. The *tfUName* and *tfUPass* are parameters we need (see Figure 1).



Invalid login!

Copyright 2019 Acunetix Ltd.

Warning: This forum is deliberately vulnerable to SQL Injections, directory traversal, and other web-based attacks. It is built using ASP and it is here to help you test Acunetix. The entire content of the forum is erased daily. All the posts are real-life examples of how attackers are trying to break into insecure web applications. Please be careful and do not follow links that are posted by malicious parties.

Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	POST	testasp.vulnweb...	Login.asp?RetURL=/Default.asp?	document	html	3.32 KB	3.14 KB
	POST	testasp.vulnweb.com	Login.asp?RetURL=/Default.asp?	document			
	GET	testasp.vulnweb...	favicon.ico	FaviconLoader.jsm...	html	1.22 KB (raced)	1.22 KB

3 requests | 4.36 KB / 4.53 KB transferred | Finish: 152 ms | DOMContentLoaded: 92 ms | load: 99 ms

Origin: http://testasp.vulnweb.com

```
tfUName=admin&tfUPass=password
```

Figure 1: Copying Request Body.

1.4 Wordlists

Kali Linux has a bunch of wordlists, so you can choose the appropriate one or just use rockyou.txt from `/usr/share/wordlists/`. So copy the worlist into your home directory and use *gzip* to unzip the archive. Type `cat rockyou.txt` to see how many passwords the wordlist contains.

1.5 Run the attack

Alright, now that we got all arguments, we are ready to fire up hydra. Here is the command pattern:

```
hydra –l <username> –P <password list> <Target hostname>
<service module>  <post request parameters>[/code]
```

Based on information we have gathered, our command should look something like this:

```
hydra −l admin −P /usr/share/wordlists/rockyou.txt
testasp.vulnweb.com http−post−form
"/Login.asp?RetURL=%2FDefault%2Easp%3F:tfUName=
^USER^&tfUPass=^PASS^:S=logout" −vV −f
```

Let's break down the commands:

- **-l <username>:** user name (not hard to guess), use -L <FILE> for a list of possible user names from a file.
- **-P <FILE>:** file with passwords, use -p <password> to literally use one password instead of guessing it.
- **testapp.vunlwebapp.com:** is a hostname or target.
- **http-post-form:** is the service module we use.
- **/Login.asp?...** This is {page URL}:{Request post body form parameters}:S={Find whatever on the page after succesfully logged in}.
- **v:** Verbose mode.
- **V:** Show *login:pass* for each attempt.
- **f:** Terminate program if pair *login:password* is found.

Now just run the command and watch the terminal output.

2 Task 2: Cracking LinkedIn Hashes Using Hashcat

In this section, you'll see how many hashes you can recover from the 2016 LinkedIn password breach. The LinkedIn hacker, a Russian, was sentenced in US court to seven years in jail on September 29, 2020. This breach of 177,500,189 unsalted SHA1 password hashes represents the data of all LinkedIn users as of 2012. Among these passwords, only 61,829,207 are unique.

In this lab you will only crack 500,000 hashes, so we'll do the following:

1. Download a copy of the file LinkedInHalfMillionHashes.txt. (use *wget*.
2. To get your feet wet, perform a "straight" dictionary attack using the *rockyou.txt* wordlist again. This attack will try each entry in the rockyou dataset with no permutations.
3. Read the man page for *hashcat* to make sure you know what the switches do and the nrun `hashcat --force -m 100 --remove --outfile=LinkedIn_cracked.txt LinkedIn_HalfMillionHashes.txt`
4. To see how many you cracked so far, run: `wc -l LinkedIn_cracked.txt` or number of passwords left with: `wc -l LinkedIn_HalfMillionHashes.txt`

3 Task 3: Create a Targeted Wordlist Using CeWL

CeWL (Custom Word List Generator) is a command-line tool that creates custom wordlists from a target website. This can be useful for cracking a password of an organization or individual that also has a website or social media profile. Because people often use information about themselves or their organization when creating passwords, custom wordlists can be very effective.

Imagine that you exfiltrated the following MD5 hash from a database:

cf4aff530715824c055892438a1ab6b2

You want to create a custom dictionary using the words on neurosecurity.byu.edu to see if you can crack the hash.

1. Create a custom dictionary using CeWL for the website neurosecurity.byu.edu (again check the arguments on the manpage): `cewl -v -d 2 -m 5 -w custom_dict.txt https://neurosecurity.byu.edu`

2. Check how many entries are in the *custom_dict.txt* file: `wc -l custom_dict.txt`

3. Look at the words in *custom_dict.txt*: `less custom_dict.txt`

4. Permute the words in the *custom_dict.txt* wordlist using the ”best64” rule, and append the output to *custom_dict.txt* (all one line): `hashcat custom_dict.txt -r /usr/share/hashcat/rules/best64.rule --stdout >> custom_dict.txt`

5. Check how many entries are in the *custom_dict.txt* file now: `wc -l custom_dict.txt`

6. Run Hashcat using *custom_dict* against the MD5 hash (all one line): `hashcat --force -a 0 -m 0 cf4aff530715824c055892438a1ab6b2 custom_dict.txt` Where *-m 0* signifies md5 mode, and *-a* specifies ”straight attack mode” (do not permutate the wordlist, because we already did). The password will be reported towards the top of the output in the format: *hash:password*. If you miss the output, you can view it in your potfile once you have cracked it by running: `hashcat --show cf4aff530715824c055892438a1ab6b2`

7. Confirm that you found the correct password: `echo -n ''<the plaintext password>'' | md5sum` We include the *-n* flag because otherwise, the echo command will append a newline character, which will throw off the hash.