**URL For tool and demo website code:** https://github.com/skhameedoutlook/brute\_force\_attack\_demo\_and\_tool

**(i). Attack Performed:**

Brute force attack to hack into another user’s account. Some variations of this attack are also discussed.

**Some Tools Available to perform the above attack:**

**1. Crunch -** Generates all possible word-list using optimized string generation algorithms and performs test on each string.

**2. Hashcat –** It is the most popular/expert password recovery tool. It supports 5 attack modes and 300+ unique hash algorithms. Also, this tool supports distributed computing(parallel processing) by using multiple CPU(s), GPU(s) simultaneously for faster password recovery.

**(ii). Tools Used in the demonstration:**

1. A tool with similar functionality as **Crunch** has been built in python. All code is available in the github repository link given above.

**(iii). Environment in which the attack is demonstrated(instead of topology):**

- A web-application hosted on a local-server has been built for demonstrating the attack.

- It contains 3 pages 1)Home Page, 2) Registration Page, 3) Login Page.

**(iv). Steps in Launching the attack:**

1. Find the username of the account for which you want to find the password.

2. Find the rules of the password format in the website.

3. Give the 1) login form’s action url, 2) username and 3) the password format as input to the tool.

**[Note: In the real world, we also need the hash algorithm used in hashing the passwords.]**

4. Run the tool using python tool\_name.py.

5. The tool stops running when the password has been found and displays it.

**Some observations and/or conclusions from the attack:**

1. Lexicographically larger passwords are more harder to find compared to lexicographically smaller passwords.

2. Given enough number of CPU(s)/GPU(s), finding the above password in any given time limit is possible.

3. Since this attack can leverage parallel processing, we need a more safer way to store passwords in critical applications.

**Differences between demo environment and actual environments:**

1. The passwords in the real world are stored in hash format, not as-it-is, so we have to know the hash algorithm and use as an extra input.

2. Brute-Force can be executed in parallel and hence making it very easy to find a password.

**4 Possible Solutions for safety from the above attack:**

1) *Limit the number of failed login attempts from a single IP address. - Has some Drawbacks*

2) If above limit is large, parallel processing brute force attacks could still compromise the password.

To solve this problem we can *limit number of failed login attempts from a single account*(*instead of IP address*).

*Real Usage Scenarios:* ICICI Bank uses this method. 3 failed login attempts for an account.

3) OTP. Apart from password, use an extra temporary one-time password sent to the users mobile number which has to be entered to login.

4) To prevent multiple password test attempts using a tool, we can use a captcha to make sure the user is not a tool/machine and hence preventing the attack completely. Stop reqiest-response using captchas. Stop machines ability to send a request/submit a form. Only a human can sumit a form.