## Lab 15: Conducting a Dictionary Attack to Crack Online Passwords Using Hydra

### Scenario

NovaTech Systems, a regional IT services provider handling sensitive healthcare data and internal corporate communications, has recently experienced an uptick in credential stuffing attempts and suspicious login activity targeting its employee portal. Despite having strong security controls such as firewalls, intrusion prevention systems, and enforced multi-factor authentication for critical accounts, weak or reused passwords remain a concern.

To evaluate the company’s security posture against password-based attacks, NovaTech’s cybersecurity team has initiated a controlled penetration testing exercise. The objective is to demonstrate how attackers could leverage publicly available tools to guess weak passwords and gain unauthorized access to a web-based system.

In this lab simulation, a cybersecurity practitioner will use Hydra to perform an online dictionary attack against a login page. The attack will attempt multiple password guesses from a predefined wordlist to determine if the targeted account is vulnerable. This activity will help highlight the importance of enforcing strong password policies and using multi-factor authentication.

### Solution

As a certified cybersecurity practitioner, the company's cybersecurity team assigned you a task to simulate a dictionary attack using Hydra against the target. The primary objective of this exercise is to identify the target login page URL and capture successful login attempts generated during the attack.

This demonstration will illustrate the process of launching Hydra from the terminal in ParrotOS, specifying the target login page along with the username and the path to the password wordlist, and executing the dictionary attack while monitoring the results in real-time.

By completing this lab, you will gain hands-on experience in using Hydra for an online password-cracking attempt, understand how dictionary attacks are performed, and recognize the security risks posed by weak or reused credentials.

**Note:** This site has been developed for specific types of hacking. Never use Hydra on any site, system, or network without prior permission from the owner.

|  |
| --- |
| 1. Turn on **ParrotOS** virtual machine. Open a Terminal, and execute the **sudo su** command to run programs with root privileges. Then, open the Hydra help menu with the following command as root user: **sudo hydra.**  For this lab, we will be focusing on the command line interface version of Hydra, but you can also access the GUI version of Hydra using the following command as the root user: **sudo xhydra.**    2. Type **hydra -h** to get the help menu and see what kind of attacks we can run using Hydra.    3. Note the examples at the bottom of the help menu, which will provide you with a better idea of the syntax Hydra supports.    4. The site we will be targeting is the following: **http://testasp.vulnweb.com/Login.asp?RetURL=/Default.asp?**.Note that this site has been developed for hacking, and you should not use Hydra on any other site without permission from the owner. In ParrotOS, open the target site in a web browser.  A screen shot of a computer  AI-generated content may be incorrect.  A screenshot of a computer  AI-generated content may be incorrect.  5. To run Hydra against an online target like this, we first need to capture the **POST form parameters**. These parameters tell Hydra exactly how to send its requests to the correct target.  Press **Ctrl + Shift + I** to launch the browser’s **Developer Tools**. Navigate to the **Network** tab, then refresh the page using **Ctrl + F5**. You will notice several **GET requests;** these indicate your machine is requesting data from the server to display the login form.      6. Type in any random username and password on the login page, then click **Login**. In the **Network** tab, you should see a new **POST** request appear; this is your machine sending the entered data to the server. This request holds the parameters we are looking for.      7. Right-click the **POST** request and choose **Edit and Resend**. A panel will appear on the right side of the **Network** tab, displaying details about the POST request.    8. Scroll to the **Request Body** section and copy the **tfUName** and **tfUPass** parameters; these are required for Hydra.    9. For this attack, we will attempt to log in as the **admin** user. To do this, we need to select a wordlist that will be used to guess the account’s password. Open the terminal and run **locate wordlists** to view all the wordlists available in ParrotOS.    10. For our attack, we will use the **rockyou.txt** wordlist. To find its location, type **locate rockyou.txt**.  11. If the rockyou.txt wordlist file has a **.gz extension** on it, we will first need to extract the file. To do this, change directory to the wordlist directory using the following command: **cd /usr/share/wordlists.**    12. Then use the following command to extract the file: **gunzip rockyou.txt.gz**  A computer screen shot of a program  AI-generated content may be incorrect.  13. Type **ls** into the terminal after this, and you will see that the **rockyou.txt** file is now available.  A computer screen with text on it  AI-generated content may be incorrect.  We now have all the information we need and are ready to open Hydra and begin the attack.  14. Start the attack by entering the following command in Hydra:  **hydra -l admin -P /usr/share/wordlists/rockyou.txt testasp.vulnweb.com http-post-form "/Login.asp?RetURL=/Default.asp?:tfUName=^USER^&tfUPass=^PASS^:S=logout" -vV -f**  After pressing **Enter**, Hydra will launch the attack, systematically trying numerous passwords for the **admin** username to gain access.  A screen shot of a computer  AI-generated content may be incorrect.  Let us break it down step-by-step:   * **-l** specifies the username we are trying to log in as * **-P** points to the wordlist we will use to guess the password for this user * **http-post-form** tells Hydra what type of request to send to the server for the login attempt * **"/Login.asp?RetURL=/Default.asp?:tfUName=^USER^&tfUPass=^PASS^:S=logout"** is the actual request Hydra sends to the server. It will replace **USER** and **PASS** with the values provided by the **-l** and **-P** options * **-vV** displays every username and password attempt being made * **-f** stops the attack as soon as the correct username and password combination is found   15. Keep in mind that Hydra will likely not succeed in guessing the password, so you can stop the attack anytime by pressing **Ctrl + C**. This demonstration shows Hydra performing a dictionary attack on a POST request. Beyond this, Hydra can also target usernames and passwords for various services, such as **SSH, FTP, Telnet, proxy**, and more, making it a highly versatile and powerful tool to include in your toolkit.  A screenshot of a computer  AI-generated content may be incorrect. |