## Lab 5-01: Scripting with Bash

**Scenario**

A hacker breached the security of a vulnerable web server of a private company, Acme Inc., gaining access to critical customer data such as email IDs and passwords. However, the system administrator does not know the vulnerabilities the hacker exploited. The company wants cyber security specialists to forensically analyze cyber-attacks to avoid such attacks in the future.

**Solution**

The company hires you as an Ethical Hacker because they want you to hack its web server and find its vulnerabilities ethically. The hacker had performed several attacks on the web server. You must identify and exploit a web application vulnerability to gain unauthorized access and demonstrate the potential consequences. However, you will act ethically and not tamper with any data or cause harm. In this lab, we perform an attack using bash scripting on a target web server using the command-line interface of the Kali Linux operating system.

|  |
| --- |
| 1. Open Text editor in Kali Linux.    2. Once you open the text editor.    3. This is the bash statement you use in every bash scripting: **#!/bin/bash or #!/bin/sh.** To display any string value or any variable value:    We use **the echo keyword;** in this, we display the **“bash scripting”** statement**.** Save the file and open the terminal to see the output.  4. Open terminal. Use the command **ls** to see the list; you will see your file.    5. You need to give permission to file for execution. For permission, use the **chmod +x *Filename*** command    6. Now, for the execution of the file, you need to use **./** command.    **// Declaration of a variable in bash scripting**  There are no data types in bash scripting. In Bash, a variable can store numeric values, individual characters, or strings of characters. You can store variables in bash by two methods, direct and indirect, like **Name=XYZ** (assign a value directly) and **new\_name=$Name (**indirectly)    **// Take input and output in bash scripting**  In bash scripting, you can take input from the user by using the **read keyword.**    **// Conditional statements in bash scripting**  1. Statements that involve **if/else** that produces a boolean result, either true or false, are called conditions. There are several ways to evaluate conditions, including **if, if-else, if-elif-else**, and nested conditionals.    2. You can use **AND (-a) OR (-o)** in this conditional statement**.**  **// Loops in bash scripting**  1. There are three types of loops: FOR, WHILE, and **DO WHILE.** While loops, check for a condition and loop until the condition remains true. The for loop, just like the while loop, allows you to execute statements a specific number of times. Each loop differs in its syntax and usage.    2. First, you initialize the value in $i, then the length of $i, andthen the increment in **$i.**  **// Automate Scripting via cron jobs**  1. Cron is a job scheduling utility that is present in Unix-like systems. The crowd daemon enables cron functionality and runs in the background. The cron reads the crontab (cron tables) for running predefined scripts.  2. You can see the cron job using the command **sudo systemctl status cron.service**    3. These are some Cron jobs syntax.  **crontab -e:** edits crontab entries to add, delete, or edit cron jobs.  **crontab -l:** list all the cron jobs for the current user.  **crontab -u username -l:** list another user's crons.  **crontab -u username -e:** edit another user's crons.    4. By command **crontab-e,** you see this and where you can edit the new job for thefuture. |