

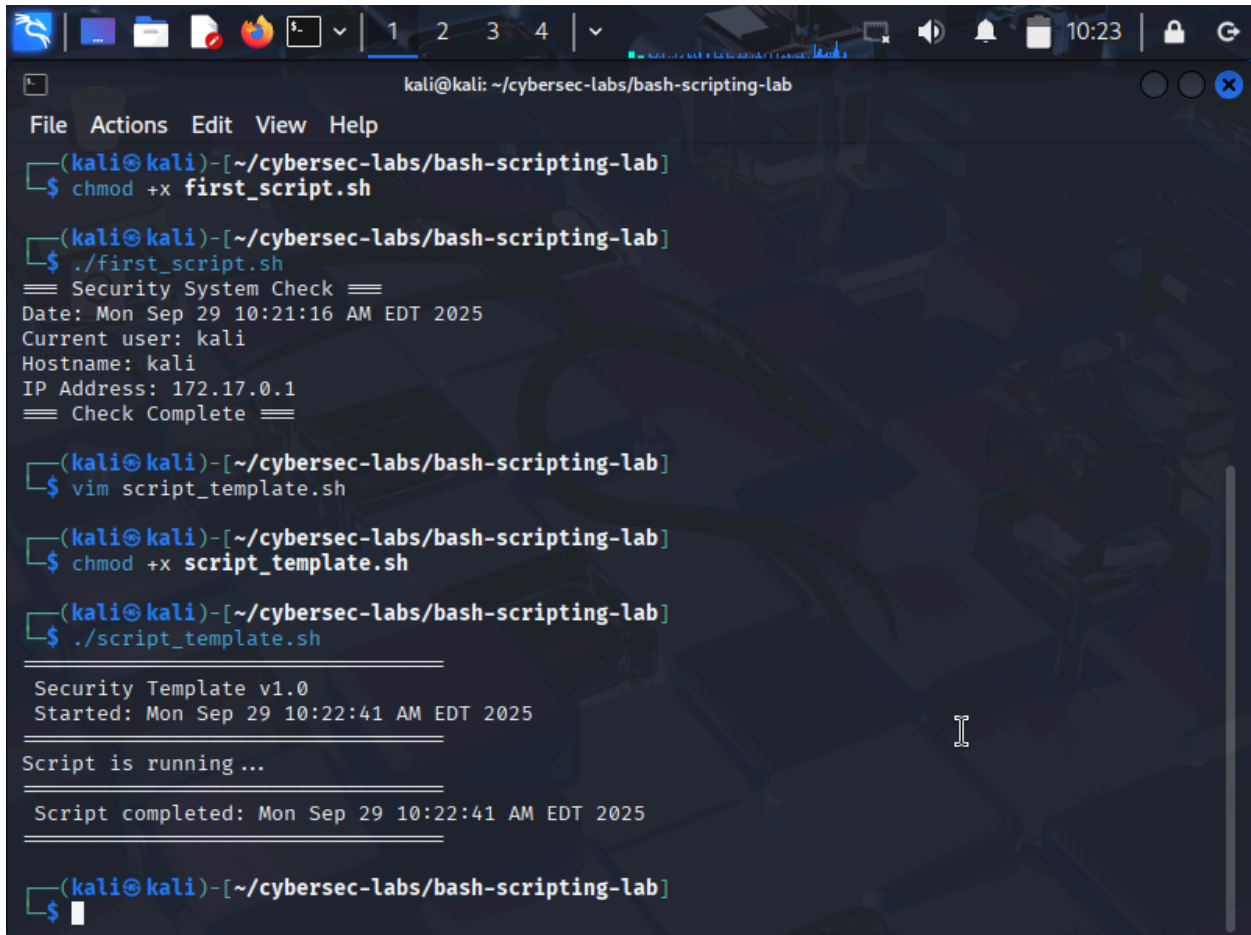
Lab 6: Linux Basics Part 6 - Bash Scripting

Matthew Cox - COMP 325

Introduction:

Bash scripting is an indispensable skill for cybersecurity professionals, serving as the foundation for automating repetitive tasks, enabling the rapid deployment of security tools, and facilitating the creation of custom utilities. Security analysts rely on scripts to automate log analysis, monitor system changes, and efficiently process large datasets, while penetration testers use them for reconnaissance automation, exploit development, and post-exploitation activities.

Body:

A screenshot of a Kali Linux terminal window. The window title is 'kali@kali: ~/cybersec-labs/bash-scripting-lab'. The terminal shows a series of commands and their outputs. The first command is 'chmod +x first_script.sh'. The second command is './first_script.sh', which outputs a 'Security System Check' with the date, user, hostname, and IP address. The third command is 'vim script_template.sh'. The fourth command is 'chmod +x script_template.sh'. The fifth command is './script_template.sh', which outputs 'Security Template v1.0', the start time, 'Script is running...', and the completion time. The terminal window has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The background of the terminal is dark with a faint, stylized image of a city skyline.

```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x first_script.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./first_script.sh
== Security System Check ==
Date: Mon Sep 29 10:21:16 AM EDT 2025
Current user: kali
Hostname: kali
IP Address: 172.17.0.1
== Check Complete ==

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ vim script_template.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x script_template.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./script_template.sh
=====
Security Template v1.0
Started: Mon Sep 29 10:22:41 AM EDT 2025
=====
Script is running...
=====
Script completed: Mon Sep 29 10:22:41 AM EDT 2025
=====

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

With the completion of the first exercise, it was a good review on how to manage nano and VIM. Completed a couple of exercises to just warm me up.

```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
Enter output filename: /tmp/scans
== Configuration Summary ==
Target: 192.168.1.100
Scan Type: tcp
Output File: /tmp/scans

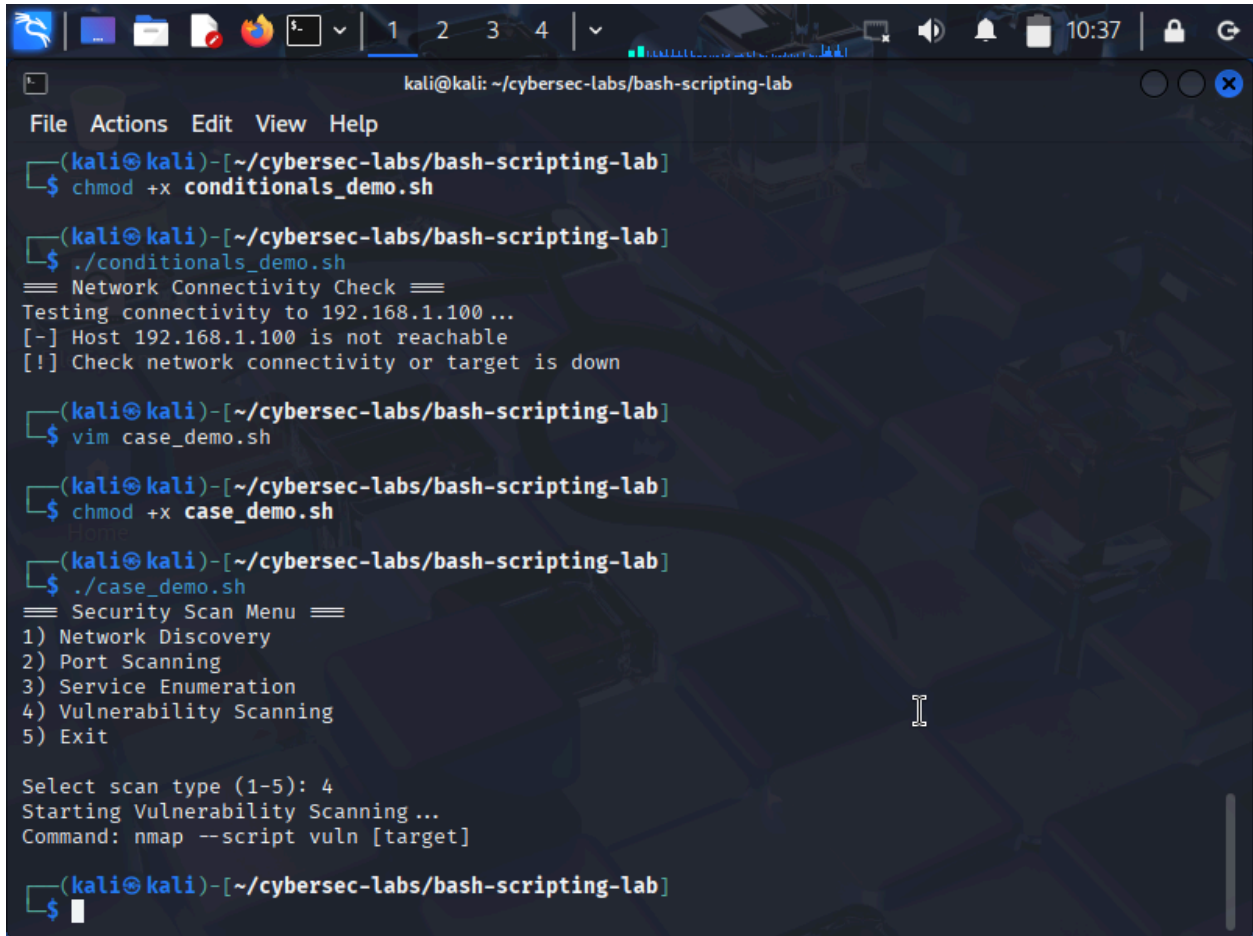
Proceed with scan? (y/n): y
Starting scan...
Scan would execute here with parameters:
  Target: 192.168.1.100
  Type: tcp
  Output: /tmp/scans
== Security Scan Configuration ==

Enter target IP address: 192.168.1.100
Enter scan type (tcp/udp): tcp
Enter output filename: /tmp/scans
== Configuration Summary ==
Target: 192.168.1.100
Scan Type: tcp
Output File: /tmp/scans

Proceed with scan? (y/n): y
Starting scan...
Scan would execute here with parameters:
  Target: 192.168.1.100
  Type: tcp
  Output: /tmp/scans

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

In exercise 2 of the lab, we executed a nano exercise with variable basics. I then made interactive scripts with VIM.

A terminal window titled 'kali@kali: ~/cybersec-labs/bash-scripting-lab' with a menu bar (File, Actions, Edit, View, Help). The terminal shows the execution of two shell scripts. The first script, 'conditionals_demo.sh', performs a network connectivity check to 192.168.1.100, which fails. The second script, 'case_demo.sh', presents a 'Security Scan Menu' with five options. Option 4, 'Vulnerability Scanning', is selected, leading to the execution of 'nmap --script vuln [target]'.

```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x conditionals_demo.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./conditionals_demo.sh
=== Network Connectivity Check ===
Testing connectivity to 192.168.1.100 ...
[-] Host 192.168.1.100 is not reachable
[!] Check network connectivity or target is down

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ vim case_demo.sh

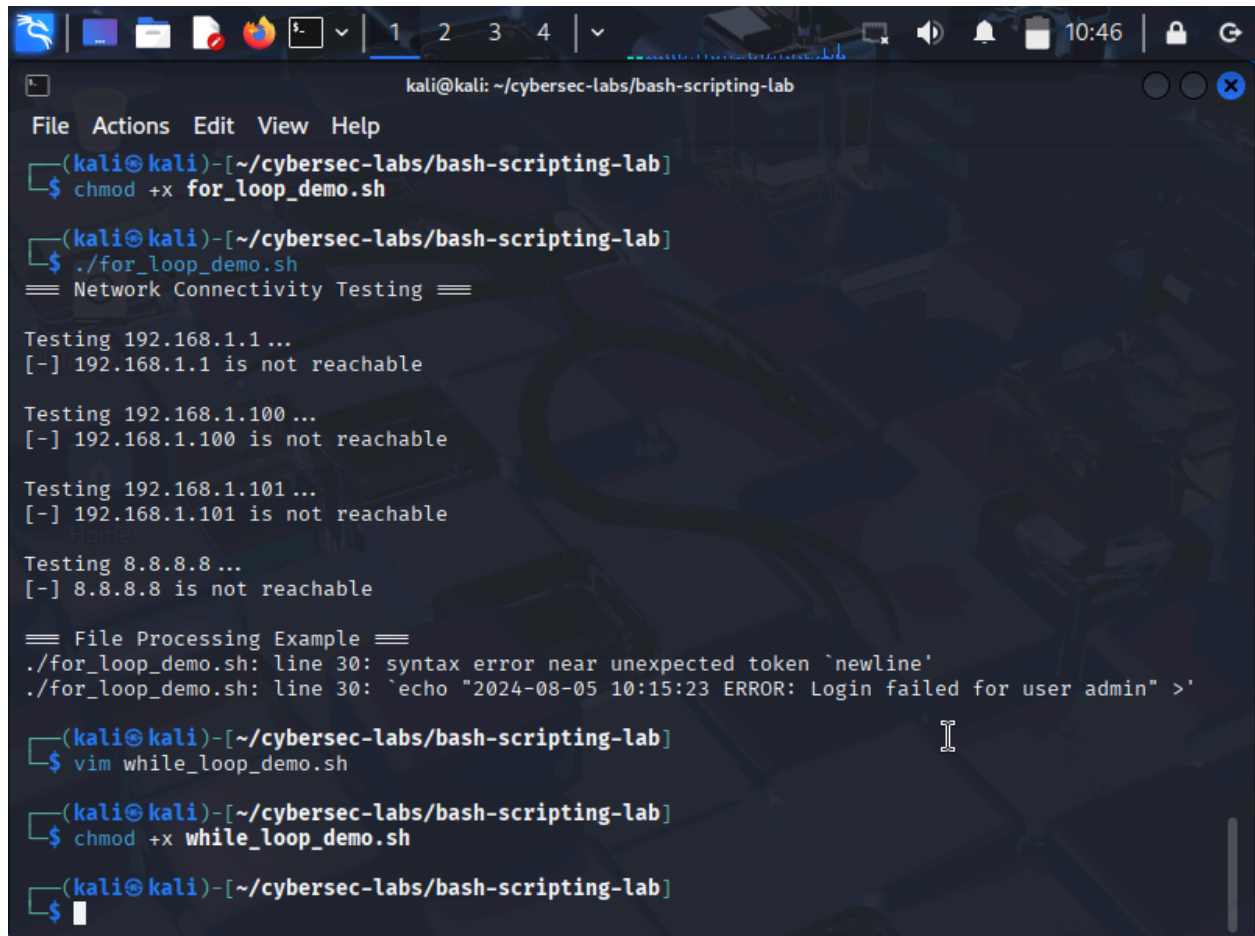
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x case_demo.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./case_demo.sh
=== Security Scan Menu ===
1) Network Discovery
2) Port Scanning
3) Service Enumeration
4) Vulnerability Scanning
5) Exit

Select scan type (1-5): 4
Starting Vulnerability Scanning...
Command: nmap --script vuln [target]

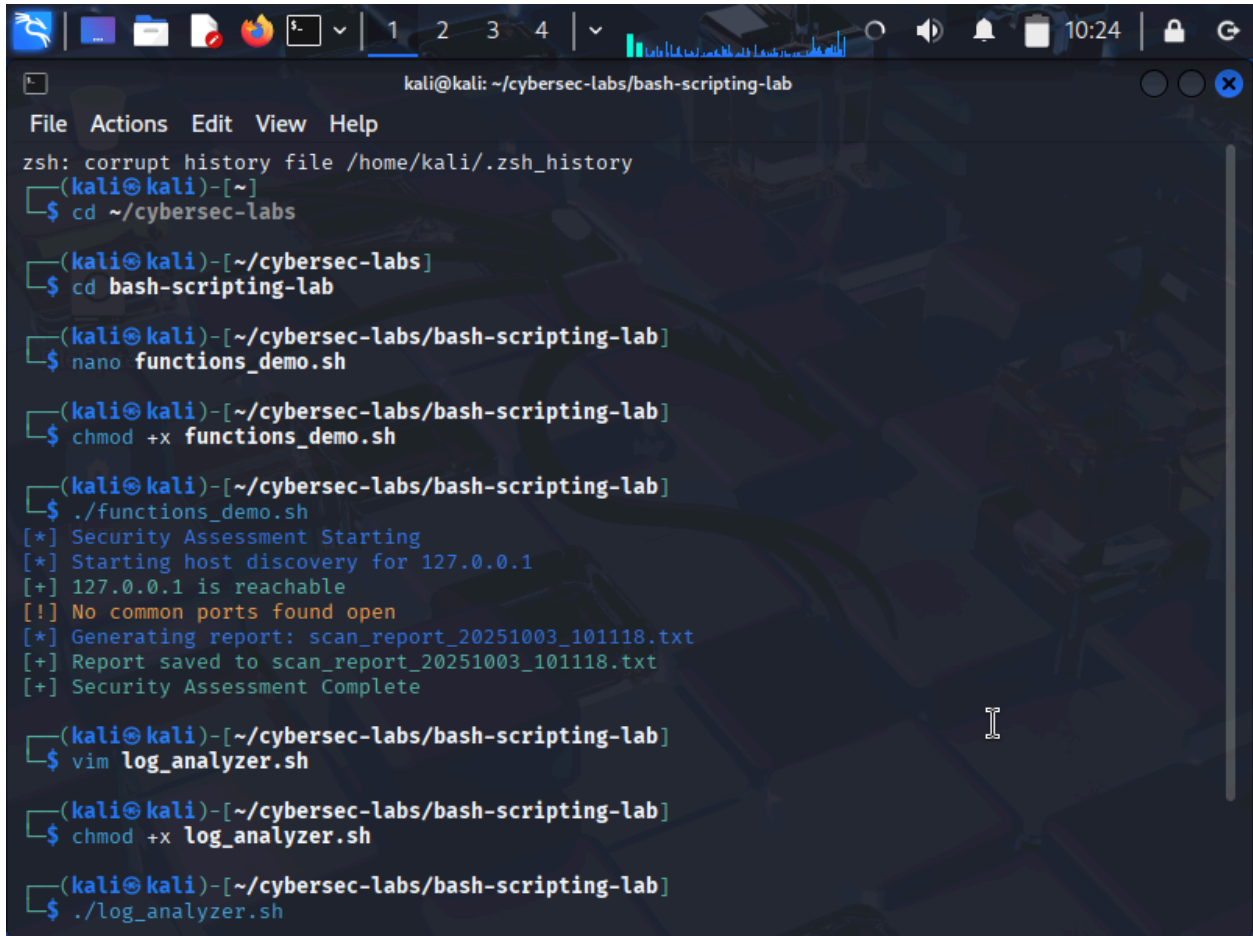
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

Another exercise with nano. This time I used if-then-else statements. Then, in the second output, I used VIM to create multiple conditions and case statements.



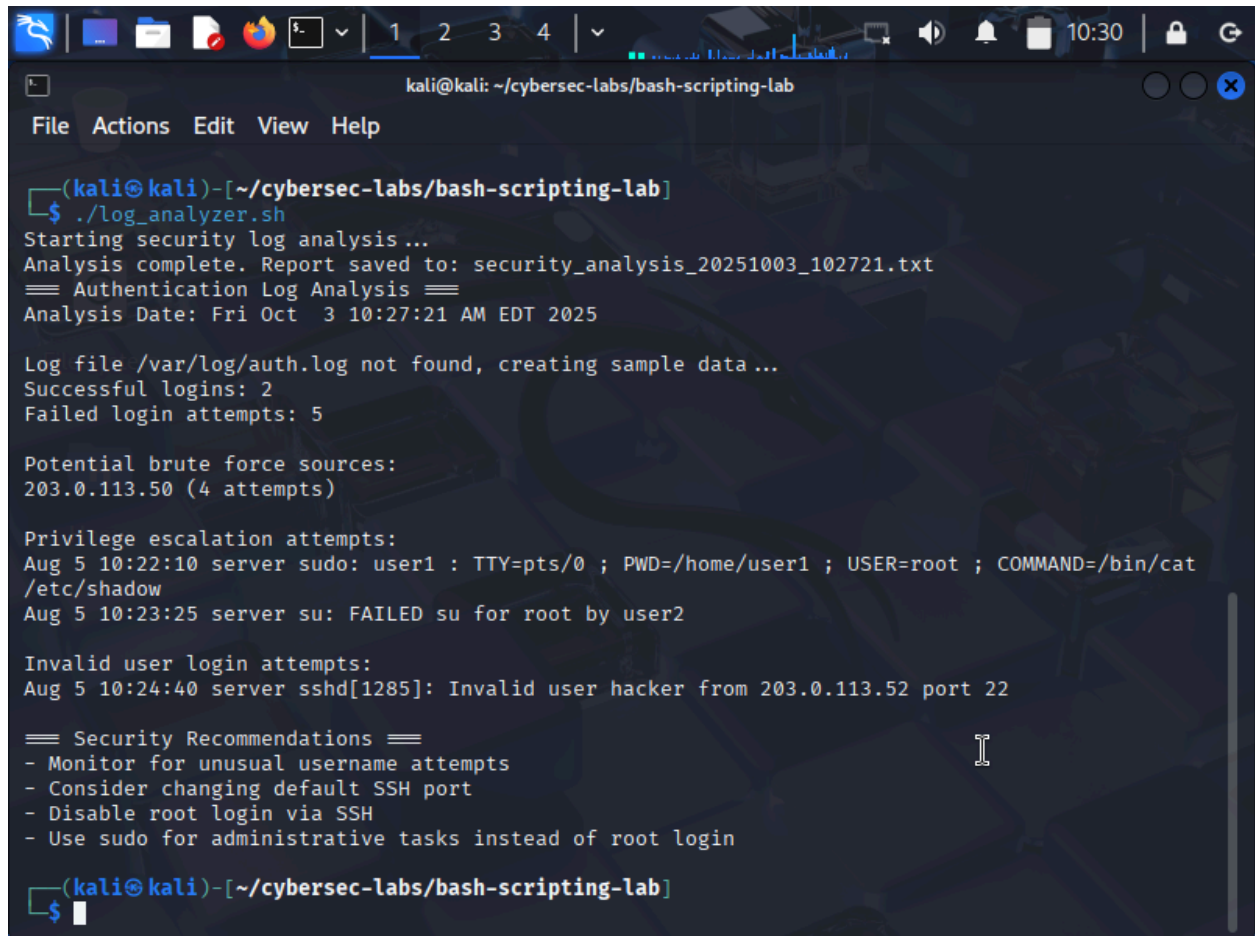
```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x for_loop_demo.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./for_loop_demo.sh
== Network Connectivity Testing ==
Testing 192.168.1.1 ...
[-] 192.168.1.1 is not reachable
Testing 192.168.1.100 ...
[-] 192.168.1.100 is not reachable
Testing 192.168.1.101 ...
[-] 192.168.1.101 is not reachable
Testing 8.8.8.8 ...
[-] 8.8.8.8 is not reachable
== File Processing Example ==
./for_loop_demo.sh: line 30: syntax error near unexpected token `newline'
./for_loop_demo.sh: line 30: `echo "2024-08-05 10:15:23 ERROR: Login failed for user admin" >'
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ vim while_loop_demo.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x while_loop_demo.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

In exercise 4, the focus was on loop automation. First, we used nano to create a for loop iteration. Following that with VIM, I create a while loop for continuous monitoring.

A terminal window on a Kali Linux system. The window title is 'kali@kali: ~/cybersec-labs/bash-scripting-lab'. The terminal shows a sequence of commands and their outputs. The user starts by navigating to the directory, creating a script, making it executable, and running it. The script performs a security assessment on 127.0.0.1. Then, the user creates another script, makes it executable, and runs it. The terminal output is as follows:

```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~]
$ cd ~/cybersec-labs
(kali@kali)-[~/cybersec-labs]
$ cd bash-scripting-lab
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ nano functions_demo.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x functions_demo.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./functions_demo.sh
[*] Security Assessment Starting
[*] Starting host discovery for 127.0.0.1
[+] 127.0.0.1 is reachable
[!] No common ports found open
[*] Generating report: scan_report_20251003_101118.txt
[+] Report saved to scan_report_20251003_101118.txt
[+] Security Assessment Complete
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ vim log_analyzer.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x log_analyzer.sh
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./log_analyzer.sh
```

In this image, I used basic functions scripts in nano. Mostly, I used it for more of an organized output.



```
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./log_analyzer.sh
Starting security log analysis ...
Analysis complete. Report saved to: security_analysis_20251003_102721.txt
== Authentication Log Analysis ==
Analysis Date: Fri Oct 3 10:27:21 AM EDT 2025

Log file /var/log/auth.log not found, creating sample data...
Successful logins: 2
Failed login attempts: 5

Potential brute force sources:
203.0.113.50 (4 attempts)

Privilege escalation attempts:
Aug 5 10:22:10 server sudo: user1 : TTY=pts/0 ; PWD=/home/user1 ; USER=root ; COMMAND=/bin/cat /etc/shadow
Aug 5 10:23:25 server su: FAILED su for root by user2

Invalid user login attempts:
Aug 5 10:24:40 server sshd[1285]: Invalid user hacker from 203.0.113.52 port 22

== Security Recommendations ==
- Monitor for unusual username attempts
- Consider changing default SSH port
- Disable root login via SSH
- Use sudo for administrative tasks instead of root login

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

This is a log analysis using Vim. This is where I started to struggle. The indentation is such a huge focus, and with large scripts like this, it is very taxing on time, trying to figure out what was wrong. This was exercise 6, and I spent the majority of my time on this.

```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./network_recon.sh -n "192.168.1.0/24"
./network_recon.sh: line 75: syntax error near unexpected token `newline'
./network_recon.sh: line 75: `echo "# This would run: nmap -sS -p 1-1000 $target" >>'

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ nano network_recon.sh

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ chmod +x network_recon.sh

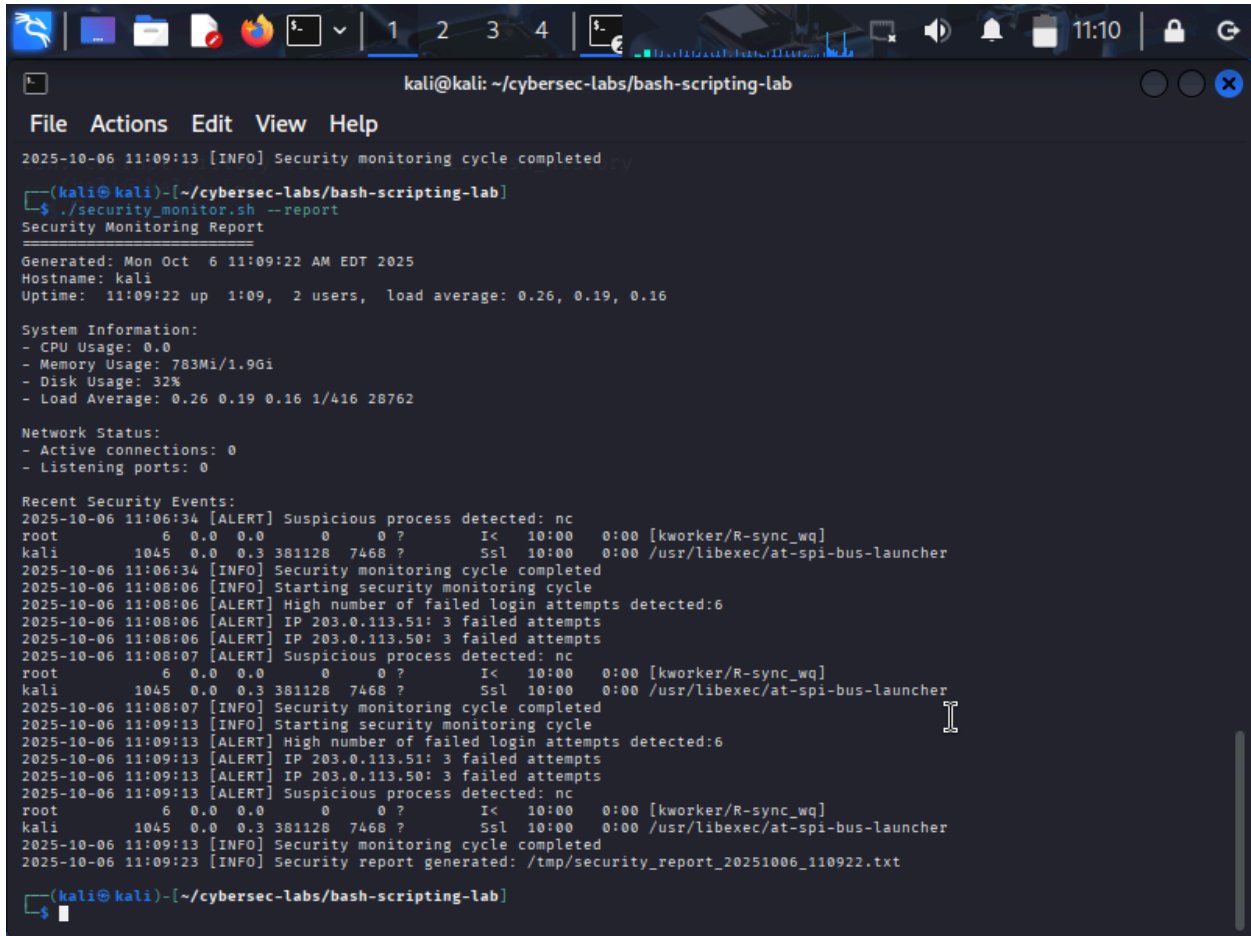
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./network_recon.sh -n "192.168.1.0/24"
== Network Reconnaissance Script ==
Target network: 192.168.1.0/24
Output directory: ./recon_results
Timestamp: 20251003_104804

[*] Starting host discovery for 192.168.1.0/24
[+] Host discovery completed. Results saved to: ./recon_results/host_discovery_20251003_104804.txt
[*] Starting port scanning
[+] Port scanning completed. Results saved to: ./recon_results/port_scan_20251003_104804.txt
ls: cannot access './recon_results/*_20251003_104804': No such file or directory
[+] Summary report generated: ./recon_results/reconnaissance_summary_20251003_104804.txt

== Reconnaissance Complete ==
Check the output directory for detailed results: ./recon_results

(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

This was exercise 7. Which was network reconnaissance using nano. I got this decently fast. I had a few indentation issues, but all I had to do was align the output text to the left. At first, it wasn't cooperating with any possible indentations.



```
kali@kali: ~/cybersec-labs/bash-scripting-lab
File Actions Edit View Help
2025-10-06 11:09:13 [INFO] Security monitoring cycle completed
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$ ./security_monitor.sh --report
Security Monitoring Report
=====
Generated: Mon Oct 6 11:09:22 AM EDT 2025
Hostname: kali
Uptime: 11:09:22 up 1:09, 2 users, load average: 0.26, 0.19, 0.16

System Information:
- CPU Usage: 0.0
- Memory Usage: 783Mi/1.9Gi
- Disk Usage: 32%
- Load Average: 0.26 0.19 0.16 1/416 28762

Network Status:
- Active connections: 0
- Listening ports: 0

Recent Security Events:
2025-10-06 11:06:34 [ALERT] Suspicious process detected: nc
root 6 0.0 0.0 0 0 ? I< 10:00 0:00 [kworker/R-sync_wq]
kali 1045 0.0 0.3 381128 7468 ? Ssl 10:00 0:00 /usr/libexec/at-spi-bus-launcher
2025-10-06 11:06:34 [INFO] Security monitoring cycle completed
2025-10-06 11:08:06 [INFO] Starting security monitoring cycle
2025-10-06 11:08:06 [ALERT] High number of failed login attempts detected:6
2025-10-06 11:08:06 [ALERT] IP 203.0.113.51: 3 failed attempts
2025-10-06 11:08:06 [ALERT] IP 203.0.113.50: 3 failed attempts
2025-10-06 11:08:07 [ALERT] Suspicious process detected: nc
root 6 0.0 0.0 0 0 ? I< 10:00 0:00 [kworker/R-sync_wq]
kali 1045 0.0 0.3 381128 7468 ? Ssl 10:00 0:00 /usr/libexec/at-spi-bus-launcher
2025-10-06 11:08:07 [INFO] Security monitoring cycle completed
2025-10-06 11:09:13 [INFO] Starting security monitoring cycle
2025-10-06 11:09:13 [ALERT] High number of failed login attempts detected:6
2025-10-06 11:09:13 [ALERT] IP 203.0.113.51: 3 failed attempts
2025-10-06 11:09:13 [ALERT] IP 203.0.113.50: 3 failed attempts
2025-10-06 11:09:13 [ALERT] Suspicious process detected: nc
root 6 0.0 0.0 0 0 ? I< 10:00 0:00 [kworker/R-sync_wq]
kali 1045 0.0 0.3 381128 7468 ? Ssl 10:00 0:00 /usr/libexec/at-spi-bus-launcher
2025-10-06 11:09:13 [INFO] Security monitoring cycle completed
2025-10-06 11:09:23 [INFO] Security report generated: /tmp/security_report_20251006_110922.txt
(kali@kali)-[~/cybersec-labs/bash-scripting-lab]
$
```

Now this exercise really tested me. This was frustrating. In this exercise, I essentially combined everything. This overall turned into a security monitoring script using Vim. Since this was a combination of the last few exercises, it had a lot of material to input. Surprisingly, indentation wasn't an issue here. But I did have a few silly mistakes, such as typos that were causing me trouble.

Knowledge Assessment:

Question 1: True or False: The shebang line `#!/bin/bash` must be the first line in a bash script.
True

Question 2: Which syntax correctly checks if a file exists in bash?
a) if [-f "\$filename"]

Question 3: True or False: In bash, the `$?` Variable contains the exit status of the last executed command.
True

Question 4: Which loop structure would be best for processing an unknown number of log files in a directory?

c) for loop with glob pattern

Question 5: What does the command `chmod +x script.sh` accomplish?

b) Makes the script executable

Conclusion:

By learning to use variables, conditionals, loops, and functions to create modular, well-documented scripts, I've moved beyond simply executing commands. I can now develop and execute sophisticated scripts for reconnaissance, threat detection, and system administration, applying these core concepts directly to real-world security challenges. Proficiency in Bash scripting provides the agility and power to secure systems more effectively. These fundamental scripting skills are the bedrock for automating security tasks, creating custom tools, and elevating my operational capabilities in any cybersecurity role.