ASSIGNMENT

LEVEL 5
COMP50001: Commercial Computing II

IF2321CS

Hand Out Date:
Hand in Date: 21st of September
Group Assessment – 24th week

Group Members:-

CB010225 - Santhusha Pansilu Kumarapeli Senanayake
CB010220 - Polpagoda Gamage Bhagya Bhavini
CB009508 -Kalinga Mudalige Rehasha Tharushi Remona Perera

INSTRUCTION TO CANDIDATES

1. **Late submission will be awarded zero (0) unless extenuating circumstances (EC) are upheld.**
2. **Cases of plagiarism will be penalized.**
3. **The assignment should be submitted as softcopy via LMS**

Acknowledgement

In order to properly acknowledge lecturer Krishnadeva for his excellent advice and assistance during the duration of this study, please accept our deepest appreciation. The success of this project has greatly benefited from the cybersecurity and Agile methods expertise of lecturer Krishnadeva. His dedication to encouraging a creative and collaborative learning environment has motivated all of us to work toward achievement. We are very grateful to him for sharing his experience and thoughts since they have greatly impacted how this project developed.

Table of Content

Contents

General Introduction

About the Application

Cybersecurity is a crucial guardian of an organization's most valuable assets in a period where the digital world is filled with ever-emerging dangers. Our team began working on a task to improve the security posture of Company XYZ's Windows 10 systems after seeing the urgent requirement. The result is a multifaceted security application methodically created in Python that improves the durability and flexibility of these systems.

This report details our effort, which came to be because of a thorough project. The goal was proactively finding and fixing vulnerabilities and misconfigurations while matching our work with generally accepted industry standards. We made the decision to follow the Center for Internet Security (CIS) guidelines, utilizing their best practices to guide the development and usability of our application.

About CIS Benchmark

A well-known industry benchmark, the CIS (Center for Internet Security) benchmark offers best practice guidelines for securing networks, software, and computer systems. It provides a thorough framework of security suggestions and configurations intended to improve the resilience to cyber threats of information technology infrastructures. Organizations can improve their cybersecurity posture, minimize vulnerabilities, and conform to accepted security standards by adhering to CIS standards. These benchmarks are a useful tool for organizations looking to proactively protect their digital assets and sensitive data because they cover a variety of security-related topics, such as operating systems, databases, web servers, and more.

CIS Benchmark Integrations

Our approach leveraged five specific controls from the CIS benchmarks, each meticulously tailored to bolster the security of Windows 10 systems. These controls, carefully selected for their relevance and impact, represent a synergy between industry-recognized standards and our commitment to proactive security.

1. Control 1: Ensure 'Enforce password history is set to '24 or more password(s)' (Automated):
    - By enforcing a password policy that requires users not to repeat their past 24 or more passwords, this control considerably improves security by avoiding password cycling.

2. Control 2: Ensure 'Maximum password age is set to '365 or fewer days, but not 0' (Automated):
    - Setting a maximum password age policy, requiring password changes within 365 days or fewer, promotes regular password updates, reducing long-term security risks.
3. Control 3: Ensure 'Minimum password age' is set to '1 or more day(s)' (Automated):

- Make sure users can't change their password too quickly. They must wait at least one day before changing it again. This is done automatically.

4. Control 4: Ensure 'Windows Firewall: Public: Settings: Display a notification is set to 'No' (Automated):
    - By ensuring that the Windows Firewall on the Public profile does not display notifications, this feature tries to avoid unwanted distractions and user behaviors that might compromise security.

5. Control 5: Ensure 'Windows Firewall: Public: Logging: Name' is set to '%System Root%\System32\logfiles\firewall\publicfw.log' (Automated):

    - This control instructs Windows Firewall to log its operations, which is necessary for monitoring and analyzing network traffic to improve security incident identification and investigation.

We aim to create a precedent for proactive and successful cybersecurity measures within Company XYZ by integrating these thorough controls with our Python-powered application. The report explains our approach to protecting digital assets and highlights our dedication to following industry standards in our aim of a more secure digital world.

Project Overview

Our team started this project with the only objective of enhancing the security of Windows 10 systems in response to the ongoing cybersecurity threats faced by Company XYZ. We did this by creating a solution that complies with accepted industry norms, particularly the benchmarks set by the Center for Internet Security (CIS).

We have made the strategic decision to focus our efforts on Windows 10, as it serves as the primary operating system for the majority of our employees, facilitating their day-to-day work. To enhance our security posture, we have extracted and analyzed five critical parameters from the CIS benchmarks. Our application effectively evaluates and corrects these configurations since it was created in a programming language (Python) with security features.

Importantly, our solution operates with few resources usage and effortlessly interacts with Company XYZ's current IT infrastructure. Its dependability is guaranteed by thorough testing and authorization.

Recognizing the dynamic nature of cybersecurity, we are dedicated to ongoing monitoring and improvement as we move toward deployment. This effort demonstrates our commitment to enhancing security and complying with accepted industry standards.

Lecturer Krishnadeva has our sincere thanks for his amazing advice and assistance. Our collaborative efforts to create a more secure digital environment are summarized in this report.

Objectives

1. Creating the application using industry-recognized CIS benchmark
2. Choosing Python because it is user-friendly and open source.
3. Creating the application with less resource intensive.
4. Creating an application for a Windows-compatible environment.

Sprints

Planning and Design

User Requirements

a. Technical Requirements:

- Compatibility and integration with existing systems.
- Automation of misconfiguration assessment and fixes.
- Detailed reporting capabilities.

b. Non-Technical Requirements:

- User training and documentation.
- Compliance with industry standards and regulations.
- Collaboration among IT, security, and business units.

c. Functional Requirements:

- Assessment of misconfigurations.
- Identification of a minimum of 5 misconfigurations.
- Automatic application of fixes.
- Role-based access control.
- Logging and reporting.

d. Non-Functional Requirements:

- Efficient performance without significant overhead.
- Security of the application and data.
- High reliability and availability.
- Scalability to accommodate network growth.
- User-friendly interface.
- Timely response to identified misconfigurations.
- Compliance with industry security standards, such as CIS benchmarks.

These user requirements encompass a wide range of technical, non-technical, functional, and non-functional aspects necessary to develop, deploy, and maintain the security application effectively within Company XYZ's complex network environment.

Why we choose windows 10 CIS benchmark?

- **Relevance:** Windows 10 is a widely used operating system, both in individual and enterprise environments. Therefore, understanding and implementing security controls for Windows 10 is highly relevant and applicable in practice.

- **Common Attack Target:** Windows-based systems are a common target for cyberattacks due to their prevalence. Familiarity with securing Windows 10 can help address real-world security threats effectively.

- **Hands-on Experience:** Windows 10 is readily available for lab and testing purposes, making it accessible for hands-on implementation and experimentation.

- **Broad Applicability:** The knowledge gained from implementing CIS benchmarks for Windows 10 can be applied to securing other Windows versions, as many security principles and settings are consistent across Windows OS releases.

Development

Development Language

We have chosen Python as the primary programming language for creating the application. Python is an excellent choice for this project due to its versatility, ease of use, and extensive libraries, making it well-suited for automation tasks and security-related development.

Integrated Development Environment (IDE) - For development, We used a Python-specific IDE PyCharm, which offers advanced debugging and code analysis tools to streamline the development process and ensure code quality.

Testing

Test Plan:

Our test plan for the Windows Security Enhancement Command Execution Tool consists of the following key testing activities:

1. Functional Testing:

    - We conducted functional testing to ensure that all the tool's commands and features, including Password Security Commands and Firewall Security Commands, work as expected.
    - Test cases were designed to validate successful execution and handle invalid inputs gracefully.

2. Security Testing:

    - While the test report doesn't explicitly mention security testing, it's essential to note that this aspect should be included in the test plan. Security testing can involve assessing the tool for vulnerabilities, such as SQL injection or other security risks.

3. Usability Testing:

    - Usability testing was implicitly conducted as part of the functional testing. The test cases aimed to validate the tool's user-friendliness by assessing how it handles user inputs and provides feedback.

Test Data:

To simulate various misconfigurations and scenarios, we utilized a combination of valid and invalid input data for different test cases. Test data included:

- Valid values for unique password, maximum password age, and minimum password age to test successful execution of Password Security Commands.
- Invalid values, such as non-numeric characters, to assess the tool's handling of incorrect inputs.
- Menu choices, including valid and invalid selections, to evaluate the tool's menu navigation and choice handling.

Testing Tools:

During the testing process of the Windows Security Enhancement Command Execution Tool, we primarily relied on manual testing techniques.

Manual Testing - The testing process involved manual execution of test cases by testers to evaluate the tool's functionality, usability, and security aspects. Testers interacted directly with the tool's user interface and provided inputs as per test scenarios.

Deployment and Monitoring

With the development phase done, we can now turn our attention to the important phases of deployment and ongoing monitoring. These steps are critical to the efficacy and long-term viability of our security solution.

Deployment

Our Python-based security application progresses from development to live production within Company XYZ's Windows 10 environment during the deployment phase. This step is meticulously carried out in order to avoid interruptions and ensure a flawless connection with the current IT infrastructure.

- Deployment Plan : We developed a complete deployment strategy prior to execution. The following major features were mentioned in this plan:

    - **Deployment Schedule:** A well-defined timeframe for each phase of the deployment process, including procedures for rollback in the event of unanticipated complications.

    - **Testing in Production:** Extensive testing in the production environment to confirm the functioning of the program and examine its influence on system performance.

    - **User Training:** Sessions to educate IT and security professionals about the application's usage, monitoring, and reporting features.

    - **Communication:** Transparent communication with all stakeholders about forthcoming changes and anticipated downtime.

- Seamless Integration : The program was carefully integrated with the existing Windows 10 infrastructure of Company XYZ. To avoid resource-intensive processes during evaluations, compatibility was guaranteed, and resource use was monitored.

Continuous Monitoring

Effective cybersecurity goes beyond initial implementation; it needs constant monitoring in order to recognize, respond to, and mitigate growing threats and vulnerabilities. Our continuous monitoring technique consists of several critical components:

- **Log Analysis -** We set up the program to log its operations in accordance with Control 9.3.7 of the CIS standards. These logs are critical to our monitoring efforts since they allow us to examine firewall activity and identify any abnormalities or security events.

- **Alerting Mechanisms -** Real-time warning systems were put in place to notify security and IT professionals as soon as the program identified any odd or suspicious activity. This guarantees that any hazards are addressed as soon as possible.

- **Performance Metrics -** We monitor the performance of our security application on a regular basis, evaluating resource usage during evaluations. Any operations that consume a lot of resources are marked for further study and optimization.

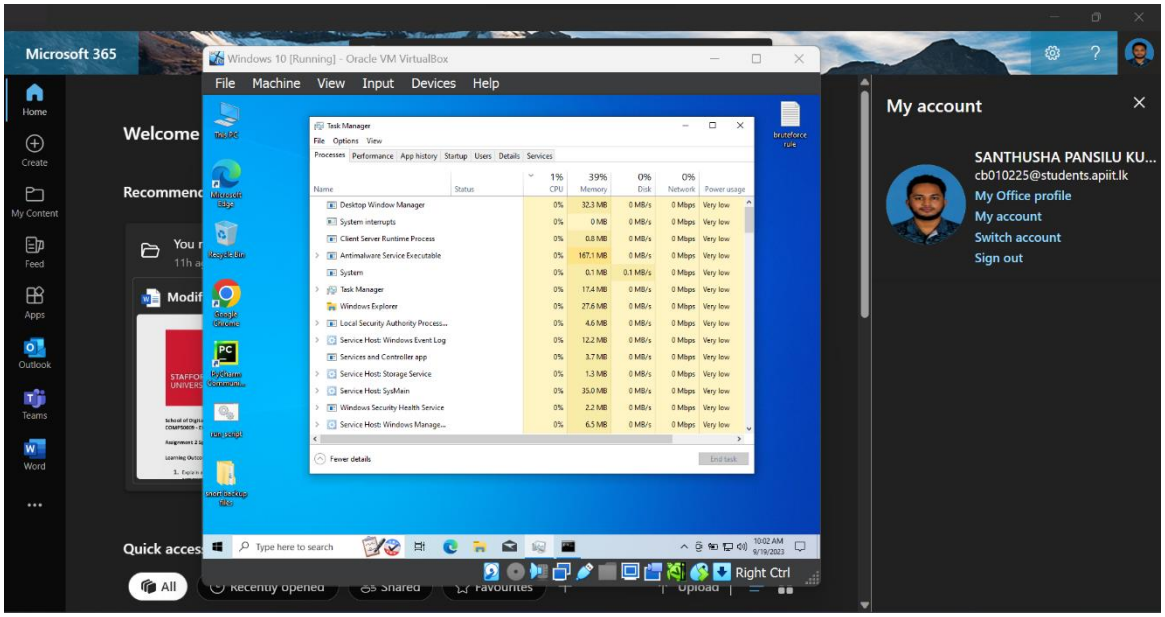
- **Regular Updates -** Our security application is subject to periodic upgrades in order to stay effective. We are dedicated to keeping up with changes in industry standards, emerging threats, and developing CIS benchmarks. To ensure top performance and security, updates and patches are implemented carefully.

- **User Feedback** - End users, IT workers, and security specialists are encouraged to submit input on the application's performance and efficacy. This recurrent feedback loop allows us to make improvements and changes over time.

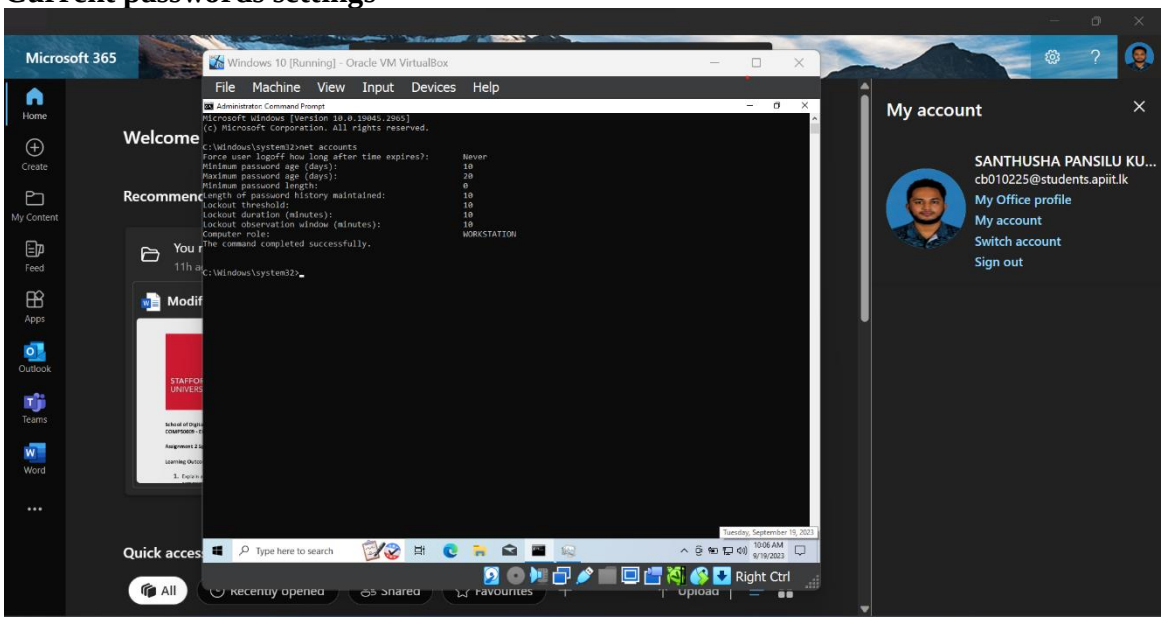
Our efforts to improve the security posture of Company XYZ's Windows 10 systems concluded in the deployment and ongoing monitoring stages. Our proactive strategy, together with adherence to industry-recognized standards, equips us to effectively defend digital assets and react to the ever-changing cybersecurity threat landscape. These phases also demonstrate our commitment to maintaining a safe digital environment and cultivating a security culture throughout the firm.

Application

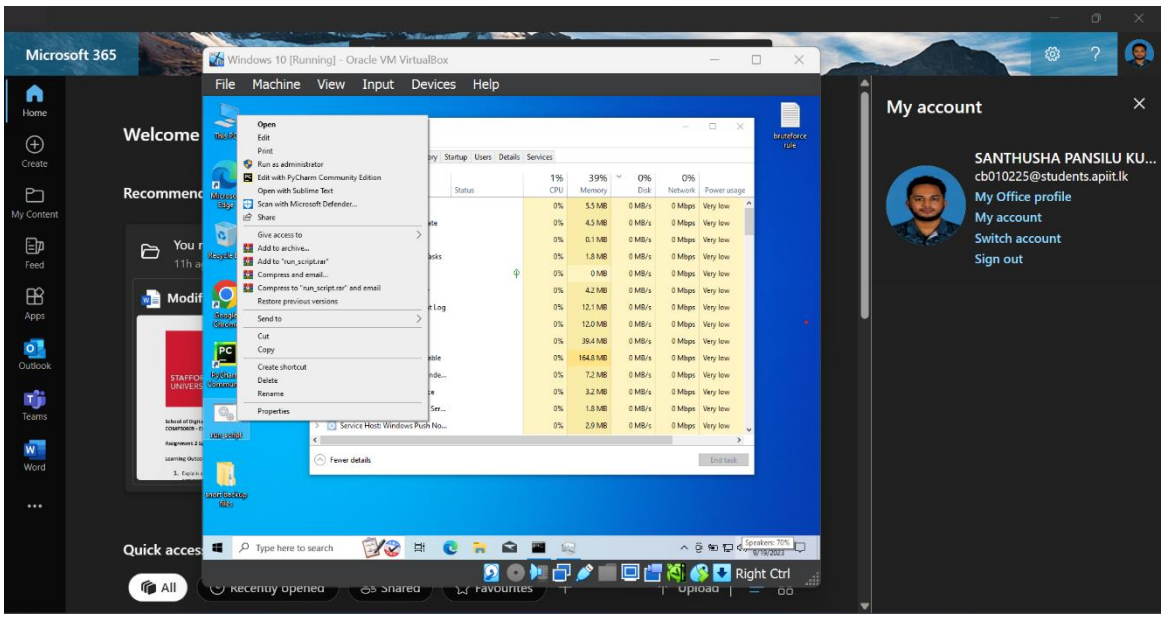
Windows 10 Machine



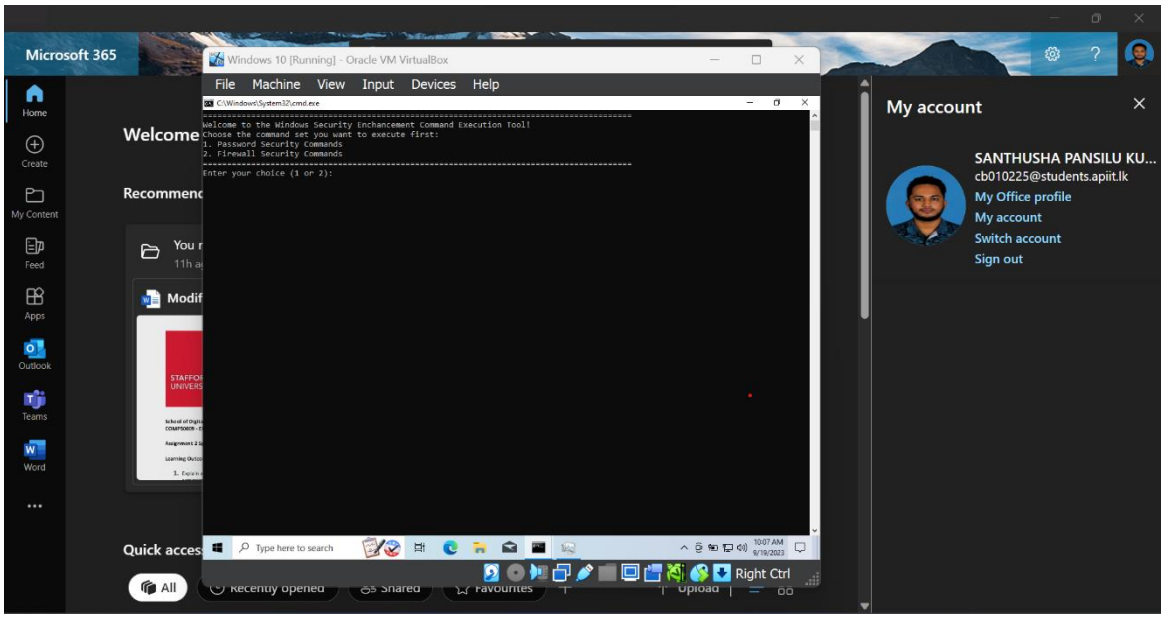
Current passwords settings



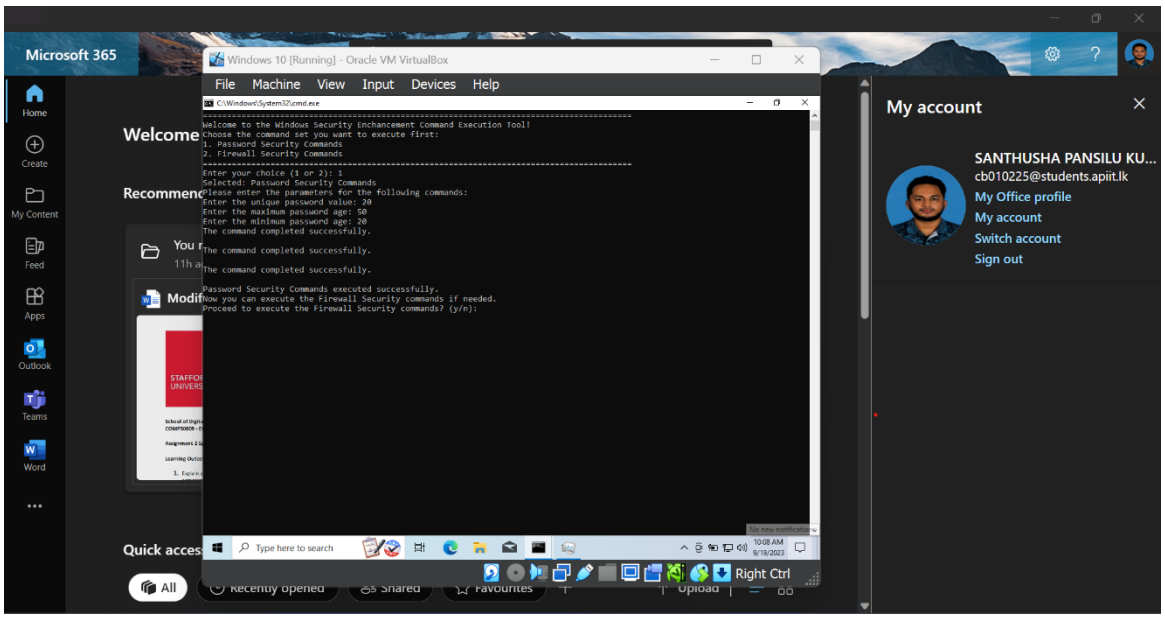
Running the application as Administrator



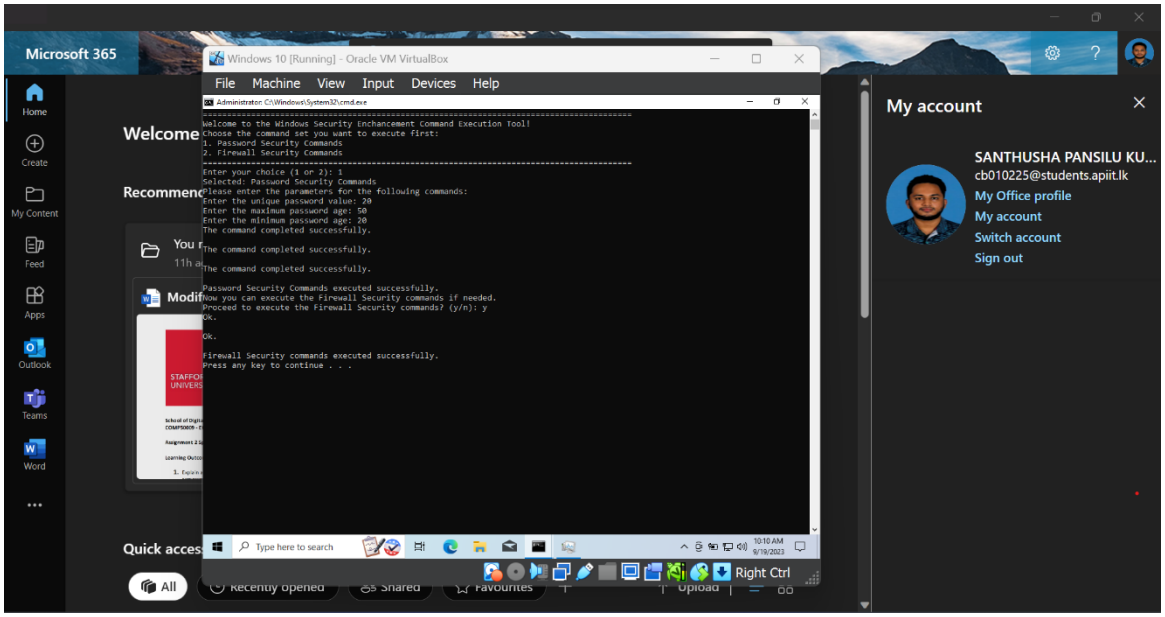
Main screen of the Application



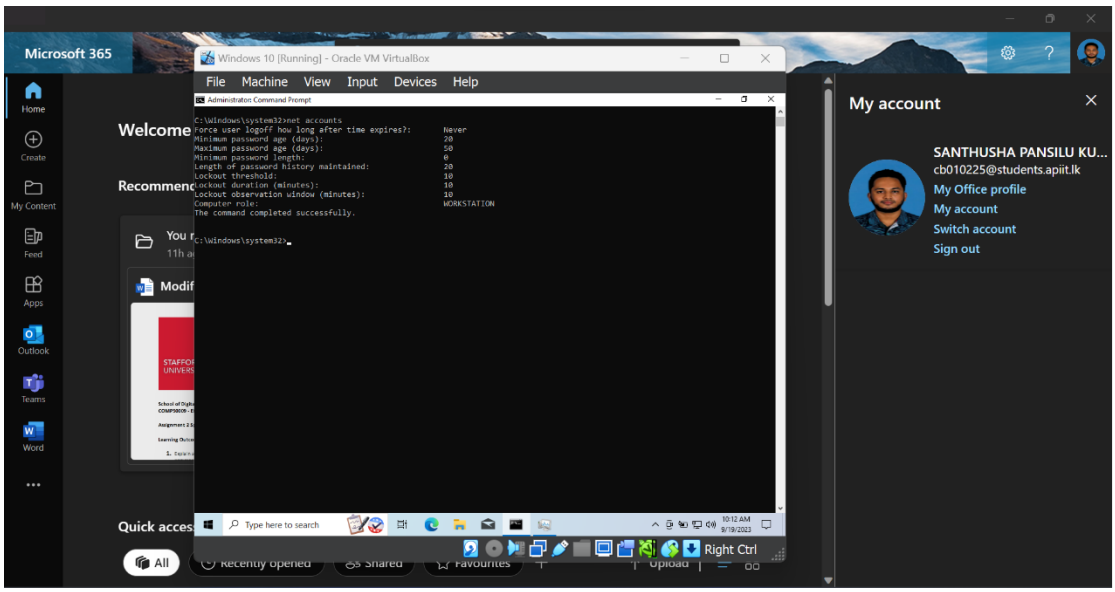
Password security enhancing command execution



Firewall security command execution



After executing the Password command this is the password settings



PART II

What is SNORT?

An open-source network intrusion detection and prevention system called SNORT was created by Sourcefire, which is now a part of Cisco. Its purpose is to monitor network traffic and look for any indication of possibly malicious activity or security risks. SNORT is frequently utilized by organizations and security experts to improve the security of their networks and systems.

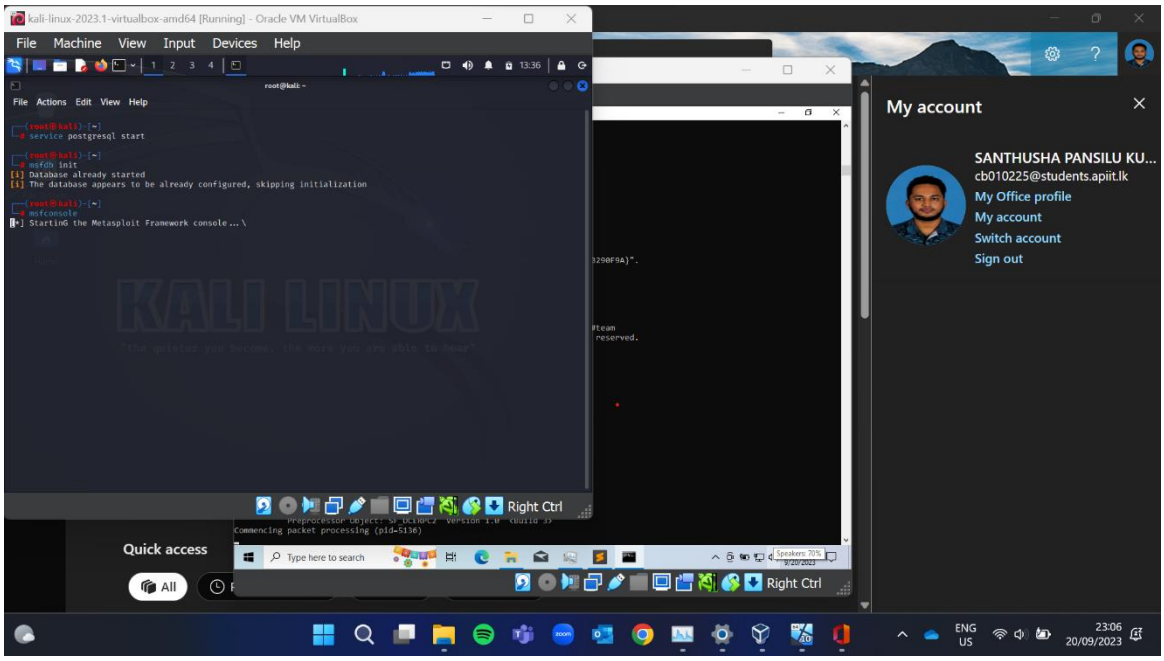
Demonstration of cyber attacks

Demonstrating the attacks by using appropriate screenshots.

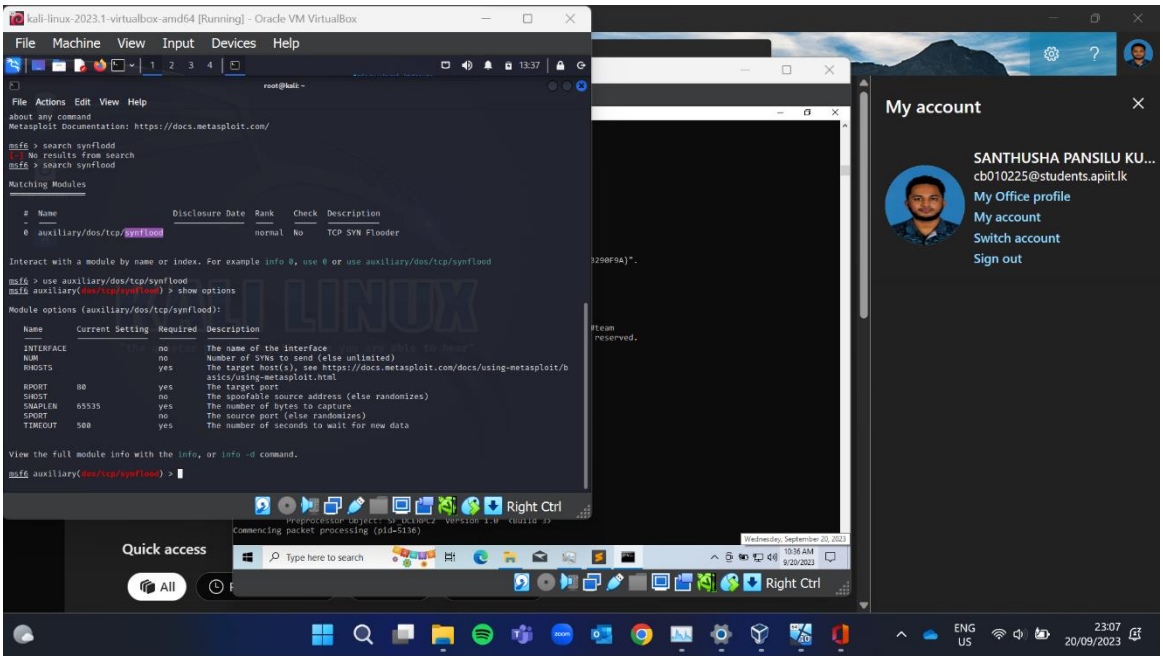
- The Attacks we choose;

1. DDOS Attack

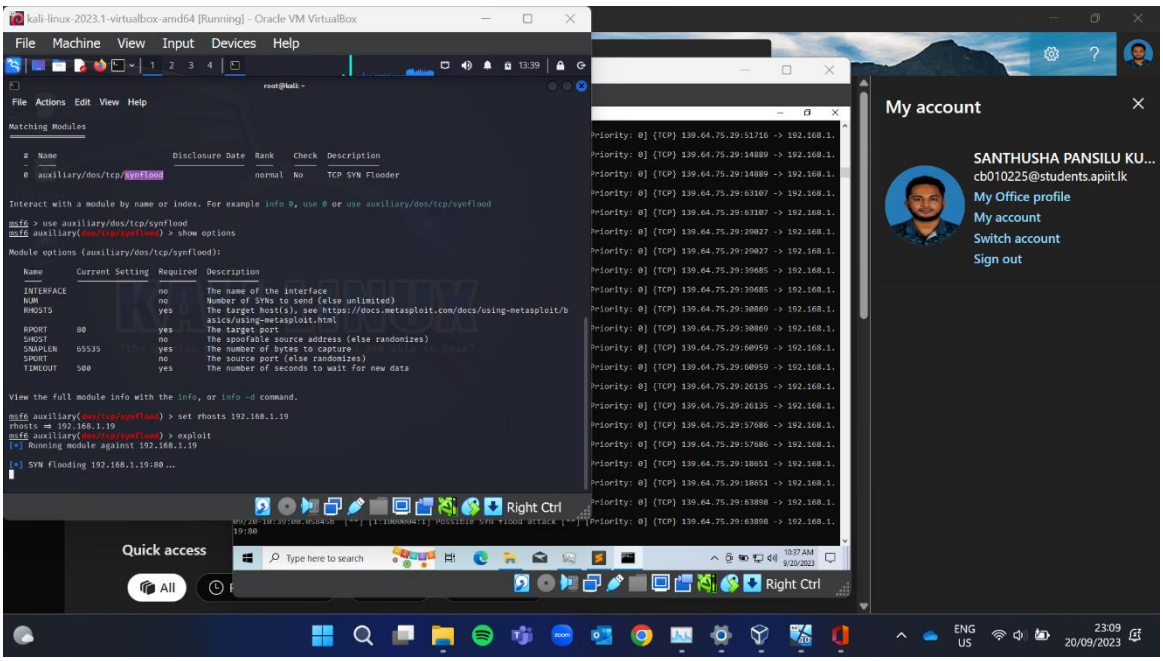
Step 01 - Initiating msf Console



Step 02 - Initiating SYN flood Attack

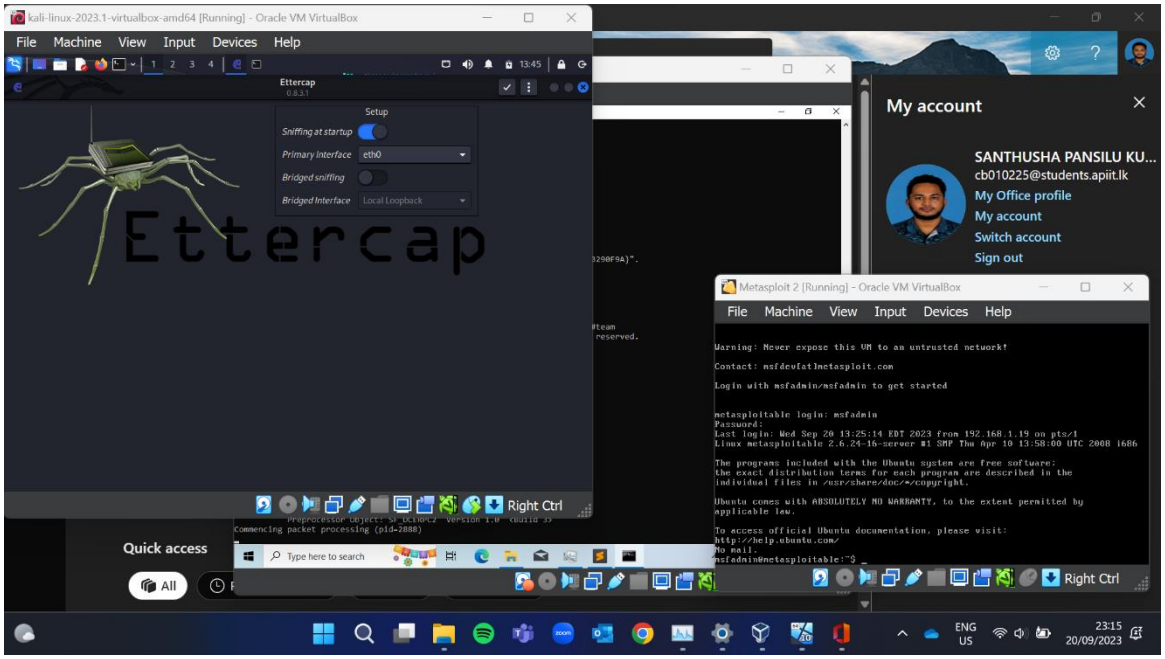


Step 03 - Executing SYN flood

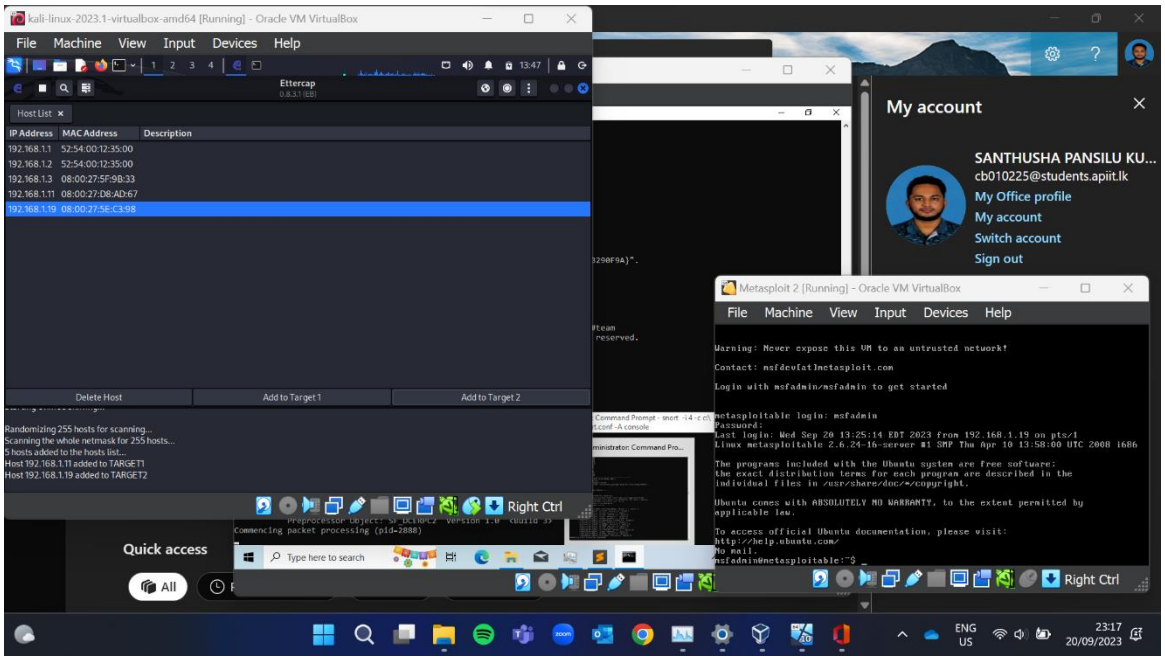


2. ARP Spoofing

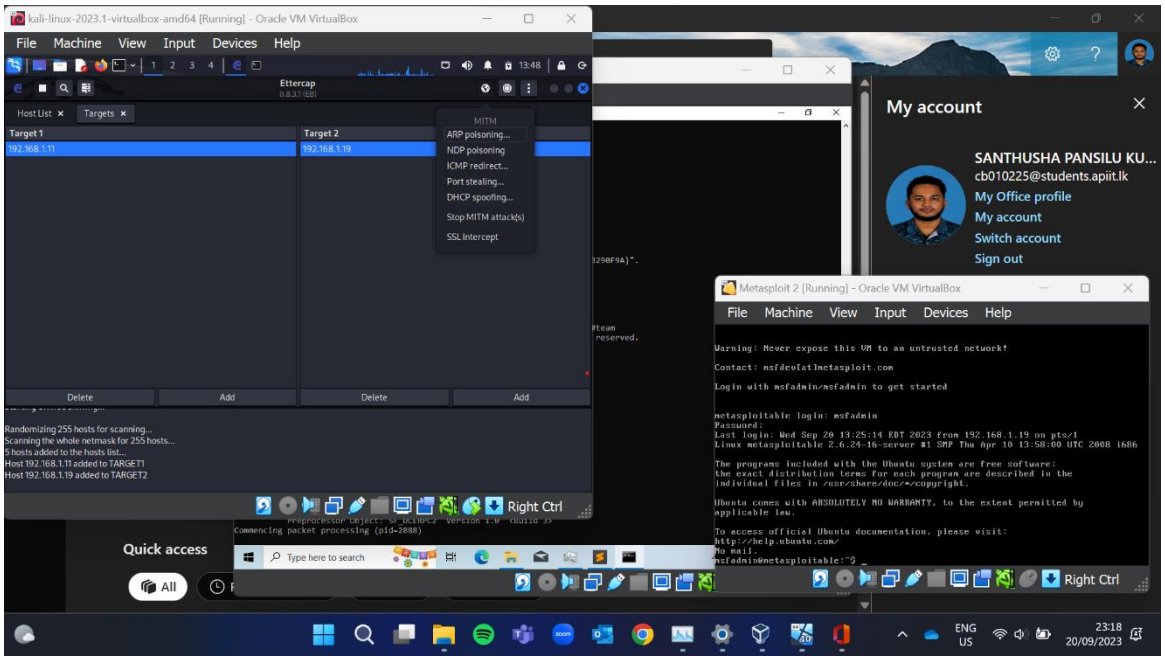
Step 01 - Initiating ARP spoofing attack



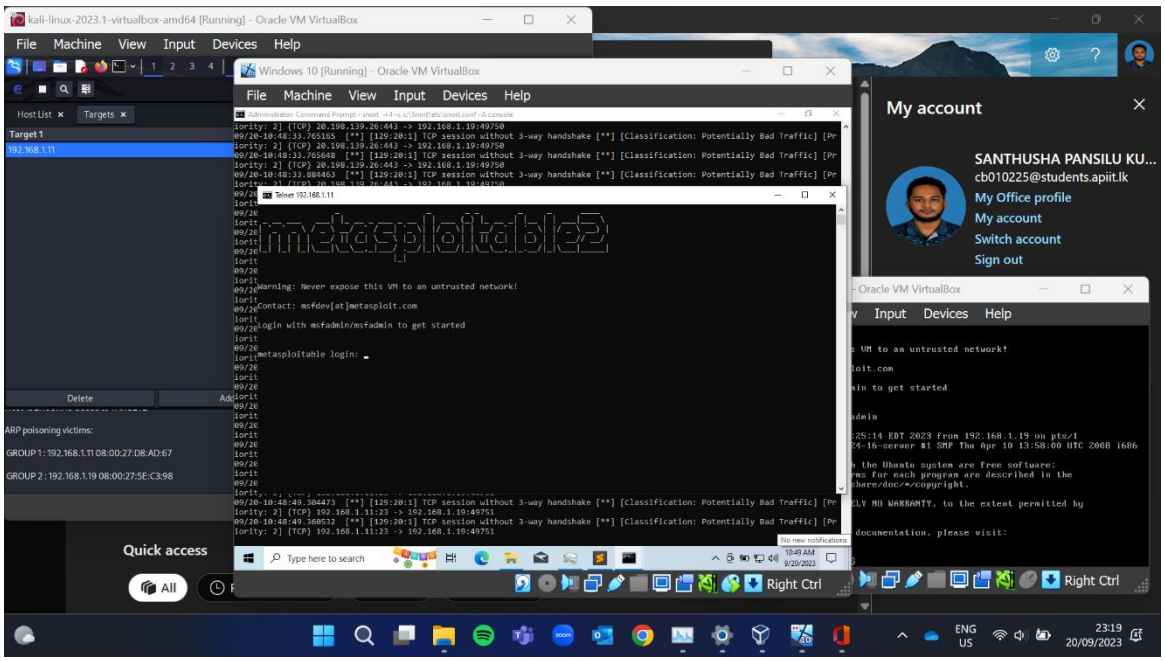
Step 02 - Selecting target hosts

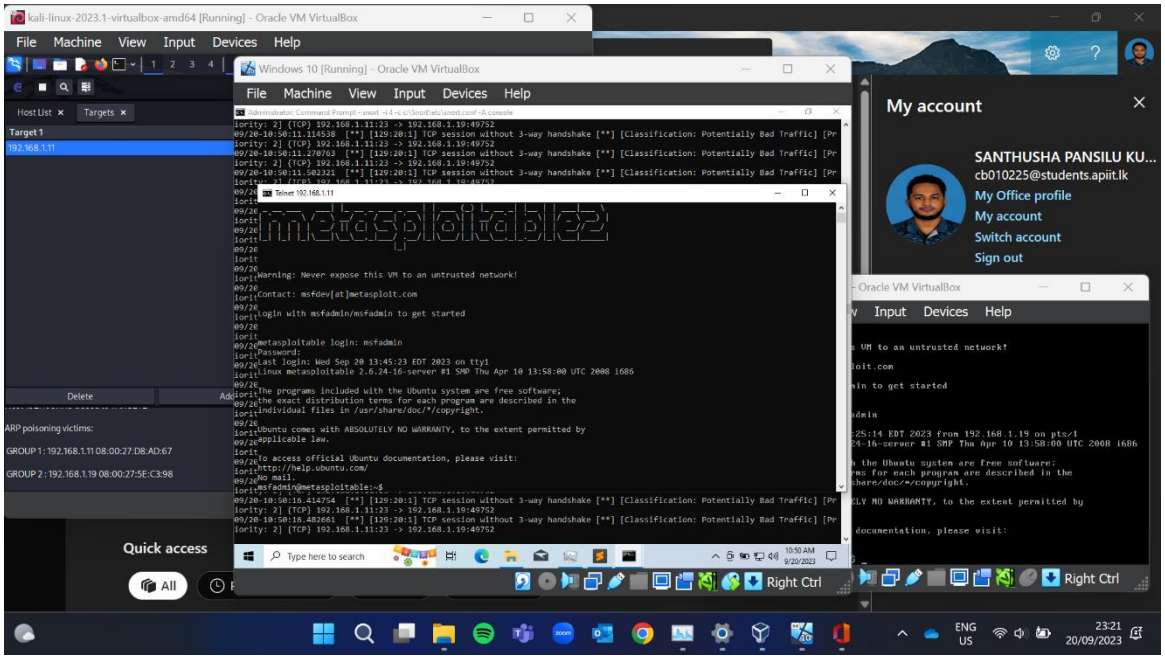


Step 03 - Starting ARP poisoning



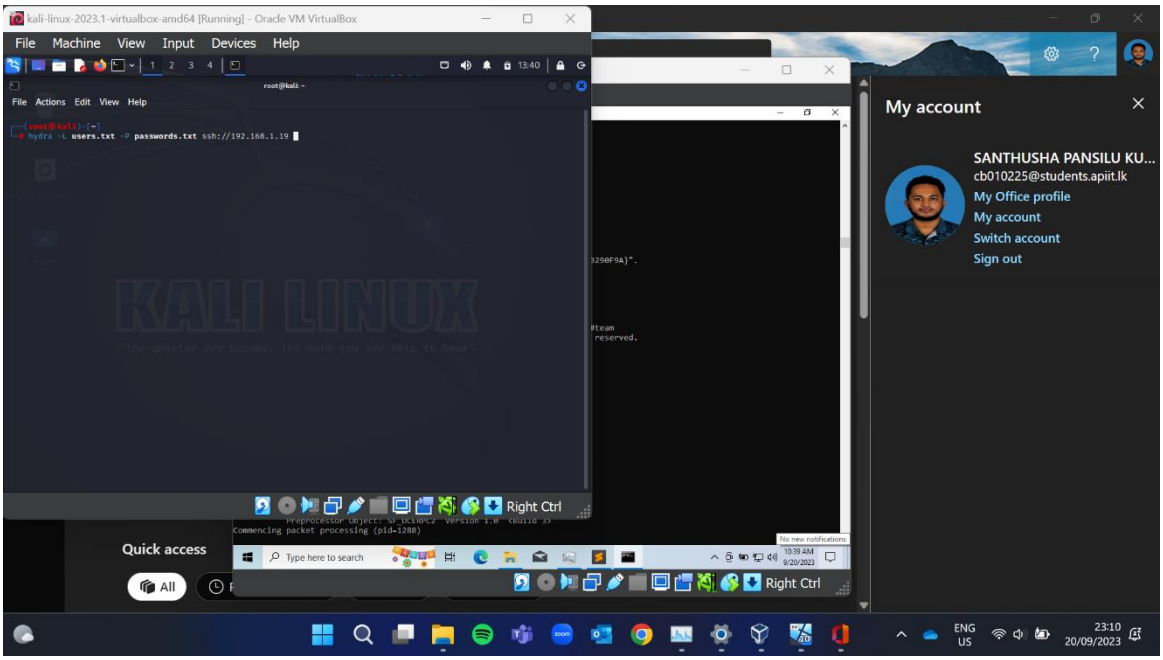
Step 04 - Telnetting Metaspoitable 2



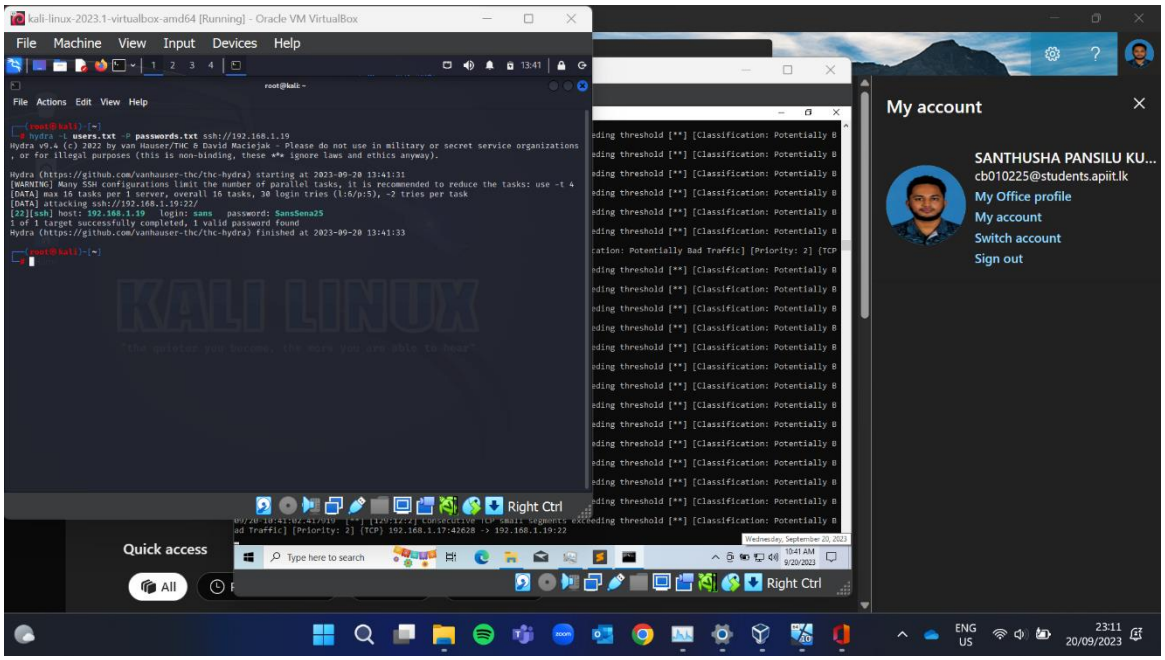


3. Bruce Force Attack

Step 01 - Initiating Brute Force Attack



Step 02 - Executing Brute Force Attack



Detecting Cyber attacks

Three snort rules to detect cyber-attacks.

1. DDOS Attack

    - alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg: "Possible SYN flood attack"; flags: S; threshold:type both, track by_src, count 100, seconds 1; sid:1000004; rev:1;)

2. ARP Spoofing

    - alert (msg: "Man in The Middle Attack"; sid: 1000006s; gid: 112; rev: 1; metadata: rule-type preproc; classtype: bad-unknown;)

3. Bruce Force Attack

    - alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg: "Potential SSH Atack"; flags: S; priority:5; threshold: type threshold, track by_src, count 10, seconds 120; classtype: suspicious-login; sid: 2001219; rev:10;)

Demonstrating the detection using appropriate screenshots.

1. DDOS Attack

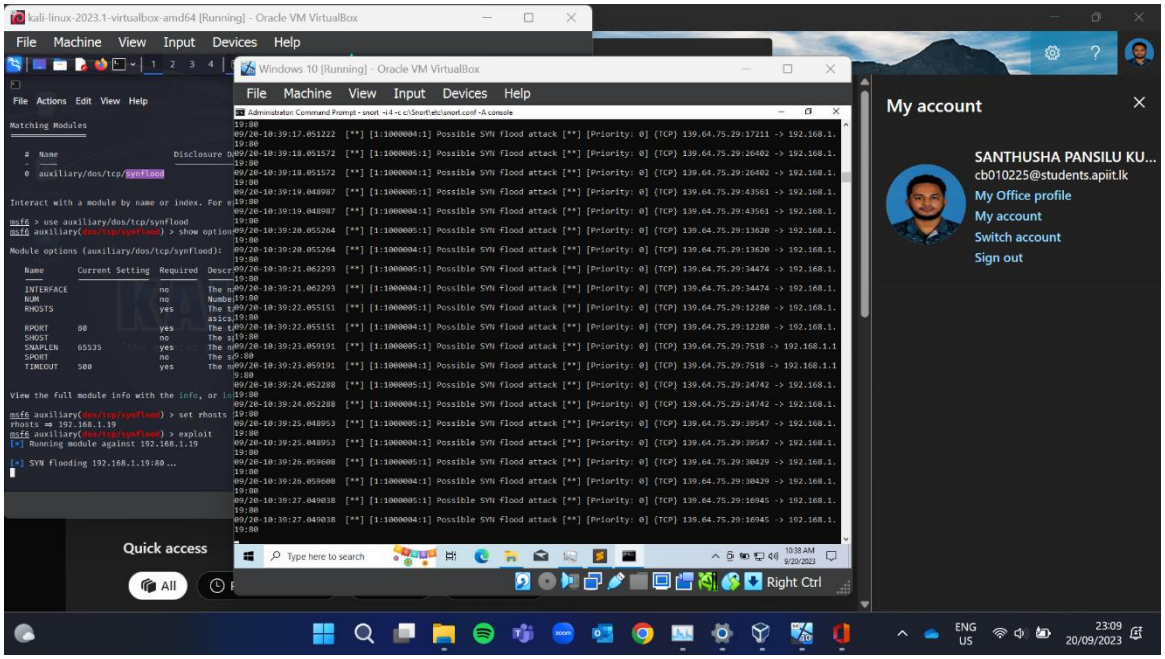


Figure 1; SNORT Detecting SYN flood

2. ARP Spoofing

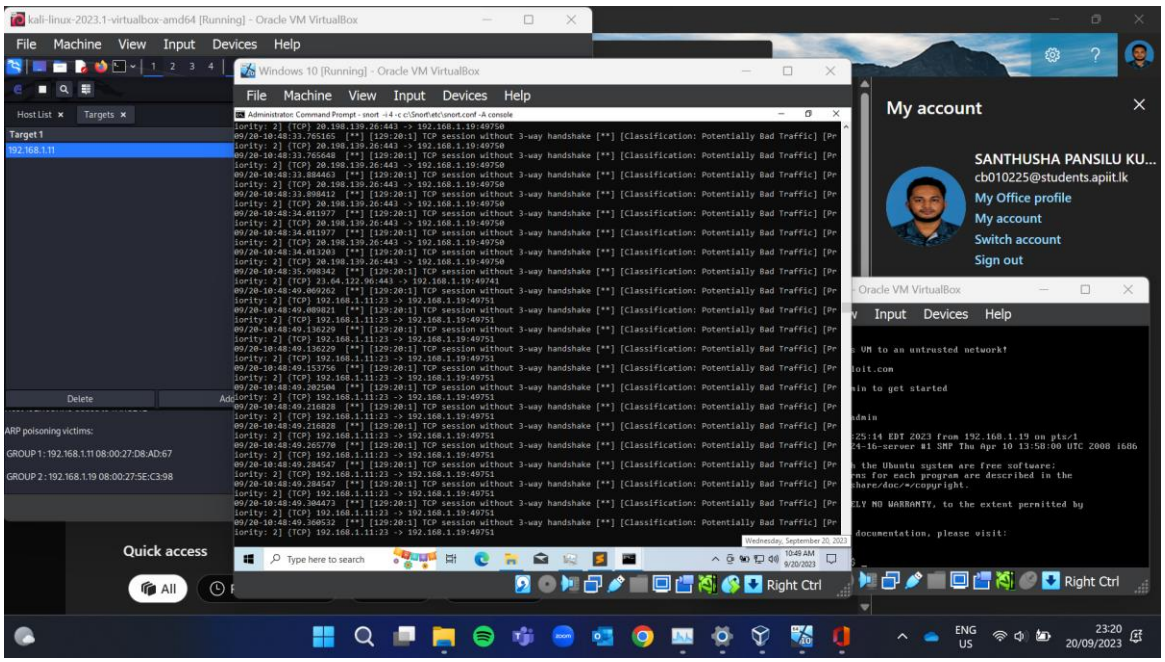


Figure 2; Detecting ARP spoofing in SNORT

3. Bruce Force Attack

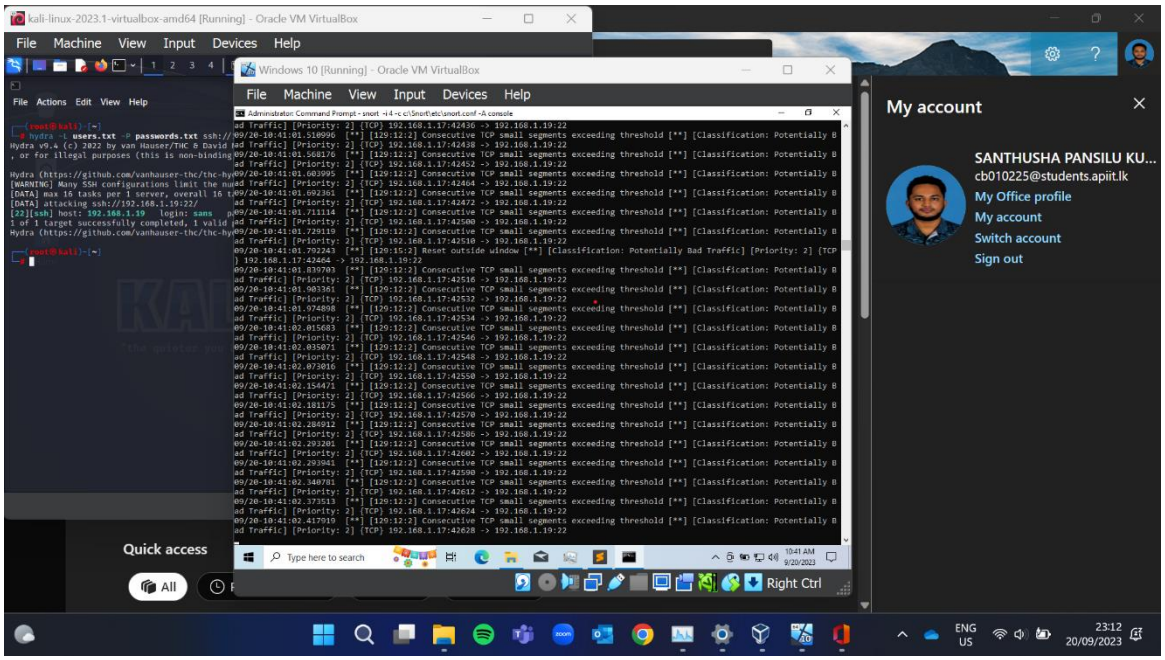


Figure 3; SNORT detecting Brute Force Attack

The effectiveness of the rules.

DDOS Attack Rule

Effectiveness: This rule is designed to detect possible SYN flood attacks, a common type of DDoS attack. It monitors incoming TCP connections and triggers an alert if it detects an abnormally high number of connection requests within a short time (100 requests in 1 second). It's effective for early detection of SYN flood attacks, allowing network administrators to take preventive measures.

ARP Spoofing Rule

Effectiveness: This rule is designed to detect potential Man-in-The-Middle (MITM) attacks, including ARP spoofing. It alerts when suspicious network activity is detected. While it's a good indicator of MITM attacks, it may also generate alerts for other network anomalies. Additional investigation is typically required to confirm the presence of an ARP spoofing attack.

Brute Force Attack Rule (SSH)

Effectiveness: This rule targets potential SSH brute force attacks. It monitors incoming TCP traffic on port 22 (SSH) and triggers an alert if it detects multiple connection attempts (10 attempts within 2 minutes) with the "S" flag set. It's effective for identifying suspicious login patterns, helping to thwart SSH brute force attacks by blocking or mitigating them.

Learning Outcomes

- **Understanding of Security Controls**: As specified in the CIS standards, gain a complete understanding of the various security controls, particularly those related to passwords and firewalls.

- **Hands-on Implementation:** Develop real-world experience putting security controls in place, especially on Windows 10 computers.

- **Cyberattack Familiarity:** Gain familiarity with popular cyberattacks, such as DDoS attacks, ARP spoofing, and brute force attacks, as well as their methods and possible consequences.

- **SNORT Rules Creation:** Learn how to create custom SNORT rules for intrusion detection and become familiar with the syntax and components of SNORT rules.

- **Attack Simulation:** Develop skills in simulating cyberattacks in a controlled environment to understand how these attacks work and their potential impact on systems.

- **Detection and Response:** Understand the importance of detecting and responding to security incidents promptly and effectively, as demonstrated by SNORT rule creation and attack detection.

- **Tuning and Optimization**: Acquire knowledge about how to fine-tune SNORT rules and security measures according to the unique requirements and features of the network environment.

- **Security Awareness**: Awareness about the importance of cybersecurity best practices and the need for continuous monitoring and improvement of security measures.

- **Real-world Application:** Gain an understanding of how security measures, attack simulations, and intrusion detection methods relate to actual situations to help to get ready for a career in network administration or cybersecurity.

Conclusion

In conclusion, our group assignment was dedicated to enhancing the security of Windows 10 systems within Company XYZ, utilizing the CIS benchmark as our foundational security standard. We meticulously implemented controls, conducted rigorous system testing, and deployed security measures to bolster the organization's cybersecurity posture. A key aspect of our approach was the deployment of SNORT, a powerful intrusion detection and prevention system. We harnessed SNORT's capabilities to detect and respond to three distinct attacks – DDoS, ARP Spoofing, and Brute Force – during simulated attacks on Windows 10 machines. This integrated solution not only fortified Company XYZ's network infrastructure but also provided invaluable experience in agile development, system testing, and security hardening. This assignment underscores the critical role of proactive cybersecurity measures in safeguarding digital assets and the importance of leveraging tools like SNORT to defend against evolving threats in today's interconnected world.

Test Report for Windows Security Enhancement Command Execution Tool

The Windows Security Enhancement Command Execution Tool is intended to provide a user-friendly interface for carrying out different security-related commands on a Windows system. There are two kinds of commands available: **password security commands and firewall security commands.** This test report assesses the tool's functioning and usefulness.

Test Environment

- Operating System: Windows 10
- Python Version: 3.8.5

Test Scenarios

Scenario 1: Execution of Password Security Commands

Test Case 1.1: Successful Execution

Test Steps:

1. Run the script.
2. Choose option 1 for Password Security Commands.
3. Enter valid values for unique password, maximum password age, and minimum password age.
4. Confirm the execution of Firewall Security Commands.
5. Proceed with the execution.

Expected Result:

- The tool should successfully execute the Password Security Commands.
- The tool should execute the Firewall Security Commands when confirmed.

Actual Result:

- The Password Security Commands were executed successfully.
- The Firewall Security Commands were executed successfully after confirmation.

Test Case 1.2: Invalid Input

Test Steps:

1. Run the script.
2. Choose option 1 for Password Security Commands.
3. Enter invalid values for unique password, maximum password age, and minimum password age (e.g., non-numeric characters).

Expected Result:

- The tool should handle invalid input gracefully and provide appropriate error messages.

Actual Result:

- The tool correctly detected invalid input and prompted for correct values.

Scenario 2: Execution of Firewall Security Commands

Test Case 2.1: Successful Execution

Test Steps:

1. Run the script.
2. Choose option 2 for Firewall Security Commands.

Expected Result:

- The tool should successfully execute the Firewall Security Commands.

Actual Result:

- The Firewall Security Commands were executed successfully.

Scenario 3: Invalid Menu Choice

Test Case 3.1: Invalid Choice

Test Steps:

1. Run the script.
2. Enter an invalid choice (e.g., a character other than "1" or "2").

Expected Result:

- The tool should handle invalid menu choices and prompt the user to try again.

Actual Result:

- The tool correctly handled invalid menu choices and prompted for a valid choice.

Both Password Security Commands and Firewall Security Commands were successfully executed using the Windows Security Enhancement Command Execution Tool. It offered suitable feedback for incorrect input and menu selections. This program is beneficial for adjusting Windows security settings because it is effective and user-friendly.