|  |  |
| --- | --- |
| **Student Number: S10198161** | **Seat Number:48** |
| **Student Name: Tan Jia Shun** | **Module Group: P02** |



**Web Application Pen-Testing**

Year 2 (2020/21), Semester 4

## School of InfoComm Technology

Diploma in Cybersecurity & Digital Forensics

**TEST 2**

Date: 24 February 2021 (Thursday)

Time: 2:30 PM – 4:00 PM

INSTRUCTIONS TO CANDIDATES:

1. Write your Student Number, Name, Module Group and Seat Number CLEARLY in the boxes provided above.
2. This paper consists of **3** pages including this cover page. Check carefully to make sure your set is complete.
3. Answer **ALL** questions.
4. Type your answers in BLUE below each question in this paper.
5. **The screenshots included in your answers must show the Host system’s (i.e., the system on which you are running your VMs) current Date and Time.**

|  |  |
| --- | --- |
| **GRADE** |  |

1. **QUESTION 1 (16 marks)**

This question is based on the following OWASP Top 10 security vulnerabilities:

* A4:2017-XML External Entities (XXE)
* A5:2017-Broken Access Control
* A6:2017-Security Misconfiguration
* A8:2017-Insecure Deserialization
* A9:2017-Using Components with Known Vulnerabilities
* A10:2017-Insufficient Logging & Monitoring

1. Match the given attack scenarios with the above listed OWASP Top 10 security vulnerabilities.

(8 marks)

|  |  |
| --- | --- |
| **Attack Scenario Description** | **Related OWASP Top 10** |
| A PHP forum uses PHP object serialization to save a “super” cookie, containing the user’s user ID, role, password hash, and other state:  **a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user"; i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}**  An attacker changes the serialized object to give themselves admin privileges:  **a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin"; i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}** | A5:2017-Broken Access Control |
| The attacker attempts to extract data from the server:  **<?xml version="1.0" encoding="ISO-8859-1"?> <!DOCTYPE foo [ <!ELEMENT foo ANY > <!ENTITY xxe SYSTEM "file:///etc/passwd" >]> <foo>&xxe;</foo>** | A4:2017-XML External Entities (XXE) |
| A major US retailer reportedly had an internal malware analysis sandbox analyzing attachments. The sandbox software had detected potentially unwanted software, but no one responded to this detection. The sandbox had been producing warnings for some time before the breach was detected due to fraudulent card transactions by an external bank. | A10:2017-Insufficient Logging & Monitoring |
| The application server comes with sample applications that are not removed from the production server. These sample applications have known security flaws attackers use to compromise the server. If one of these applications is the admin console, and default accounts weren’t changed the attacker logs in with default passwords and takes over. | A6:2017-Security Misconfiguration |

**QUESTION 1 (a) (Contd.)**

|  |  |
| --- | --- |
| **Attack Scenario Description** | **Related OWASP Top 10** |
| Shodan IoT search engine can help find devices that still suffer from Heartbleed vulnerability that was patched in April 2014. | **A9:2017-Using Components with Known Vulnerabilities** |
| The application uses unverified data in a SQL call that is accessing account information:  **pstmt.setString(1, request.getParameter("acct")); ResultSet results = pstmt.executeQuery( );**  An attacker simply modifies the ‘acct’ parameter in the browser to send whatever account number they want. If not properly verified, the attacker can access any user’s account.  **http://example.com/app/accountInfo?acct=notmyacct** | A5:2017-Broken Access Control |
| An attacker uses scans for users using a common password. They can take over all accounts using this password. For all other users, this scan leaves only one false login behind. After some days, this may be repeated with a different password. | A10:2017-Insufficient Logging & Monitoring |
| Directory listing is not disabled on the server. An attacker discovers they can simply list directories. The attacker finds and downloads the compiled Java classes, which they decompile and reverse engineer to view the code. The attacker then finds a serious access control flaw in the application. | A6:2017-Security Misconfiguration |

**QUESTION 1 (Contd.)**

1. Match the given countermeasures against the above listed (**QUESTION 1)** OWASP Top 10 security vulnerabilities.

(8 marks)

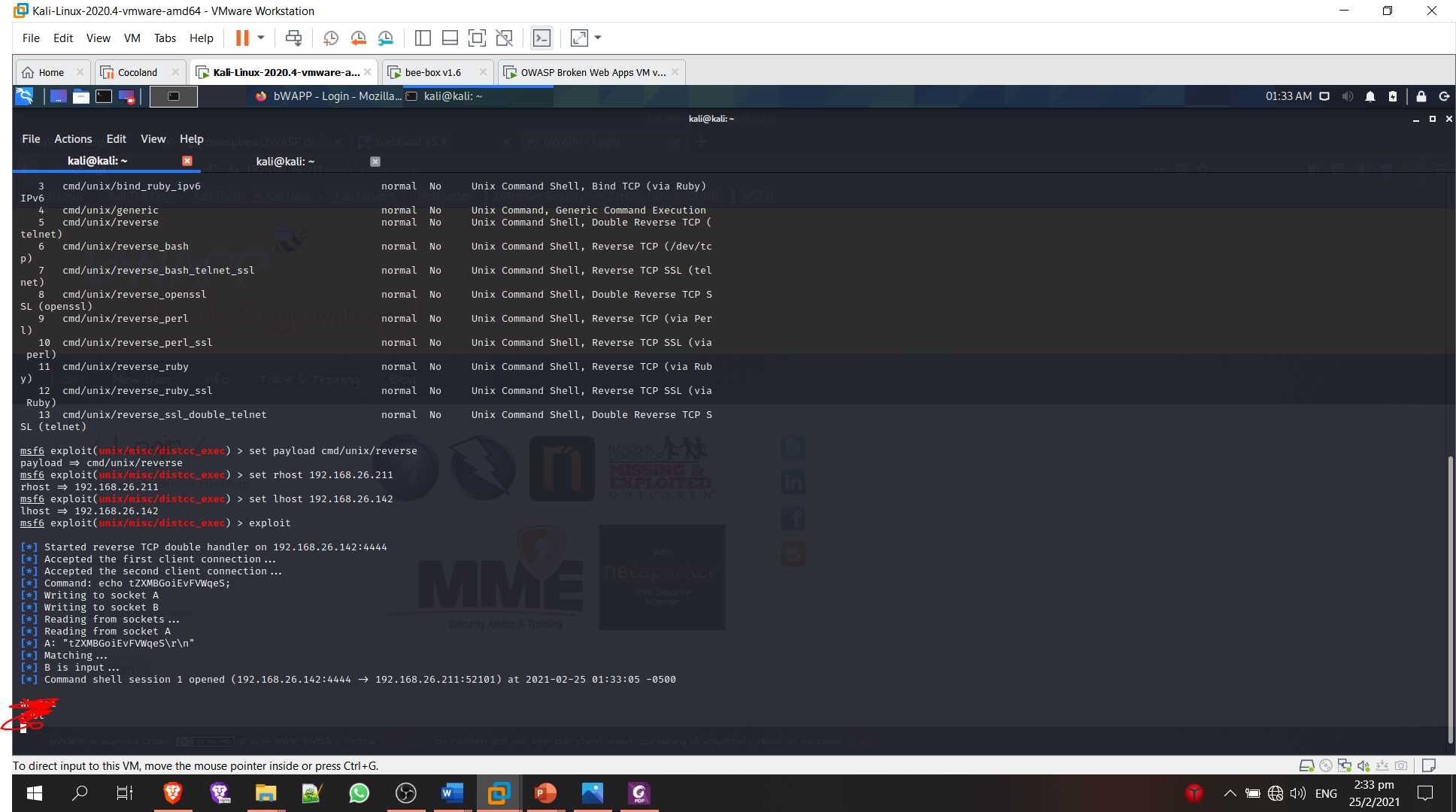
|  |  |
| --- | --- |
| **Countermeasures Description** | **Related OWASP Top 10** |
| Ensure all login, access control failures, and server-side input validation failures can be recorded with sufficient user context to identify suspicious or malicious accounts and held for sufficient time to allow delayed forensic analysis. | A10:2017-Insufficient Logging & Monitoring |
| Access control is only effective if enforced in trusted server-side code, where the attacker cannot modify the access control check or metadata. | A5:2017-Broken Access Control |
| Continuously inventory the versions of both client-side and server-side frameworks, libraries, and their dependencies. Continuously monitor sources like CVE and NVD for vulnerabilities in them. | A9:2017-Using Components with Known Vulnerabilities |
| A minimal platform without any unnecessary features, components, documentation, and samples. Remove or do not install unused features and frameworks. | A6:2017-Security Misconfiguration |
| Implementing integrity checks such as digital signatures on any serialized objects to prevent hostile object creation or data tampering. | A8:2017-Insecure Deserialization |
| Patch or upgrade all XML processors and libraries in use by the application or on the underlying operating system. | A4:2017-XML External Entities (XXE) |
| Ensure high-value transactions have an audit trail with integrity controls to prevent tampering or deletion, such as append-only database tables or similar. | A10:2017-Insufficient Logging & Monitoring |
| A repeatable hardening process that makes it fast and easy to deploy another environment that is properly locked down. | A6:2017-Security Misconfiguration |

1. **QUESTION 2 (16 marks)**

Use appropriate tools to exploit any ONE of the OWASP Top 10 security vulnerabilities indicated in **QUESTION 1** . The goals for this exploitation are as follows:

1. Successfully gain a command shell session in the server that is hosting the bWAPP, an extremely buggy web app!

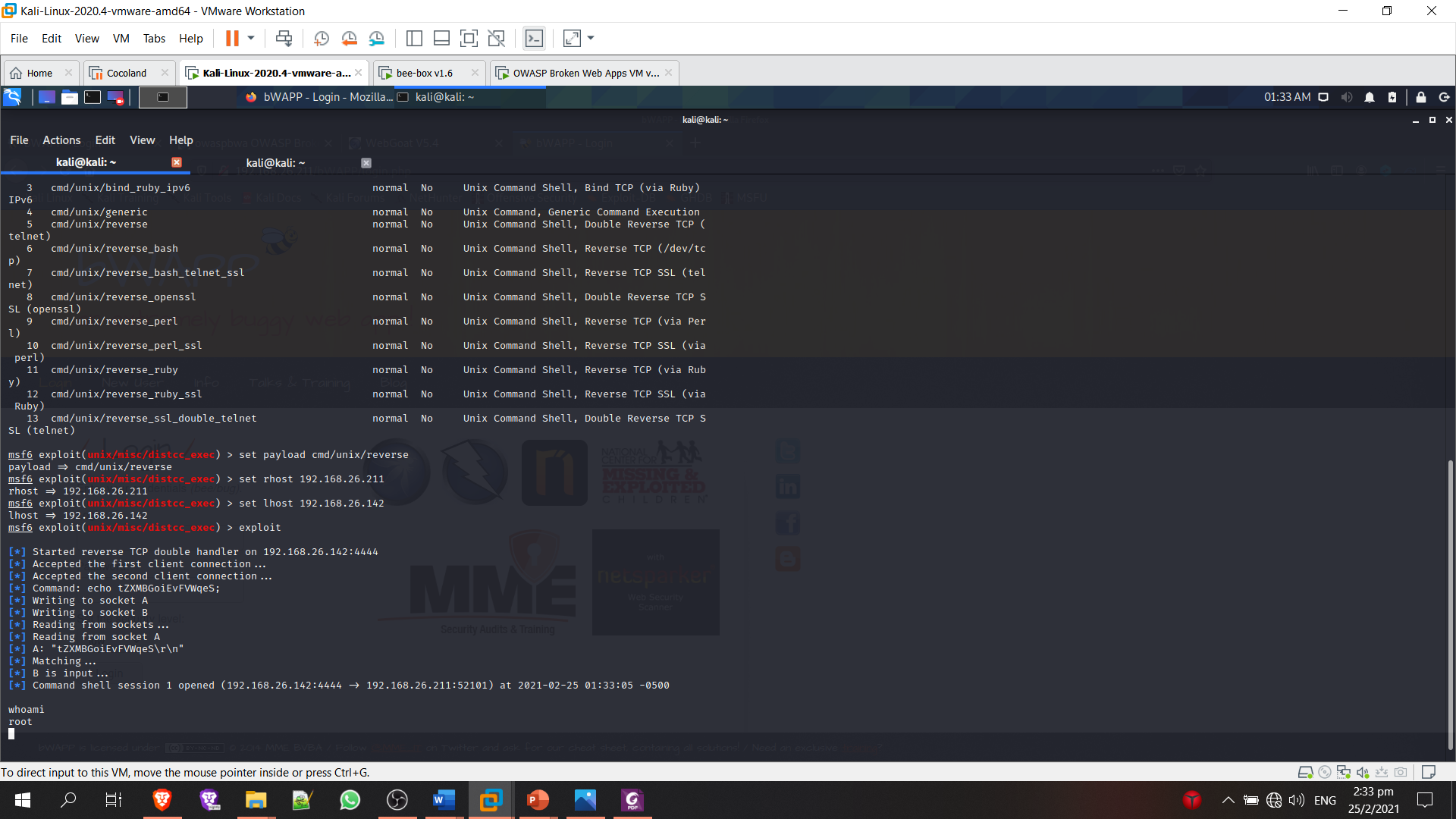
Provide detailed screenshots of the most critical steps that you took to achieve this goal, with a brief description. The screenshots must show the Host system’s (i.e., the system on which you are running your VMs) current Date and Time.



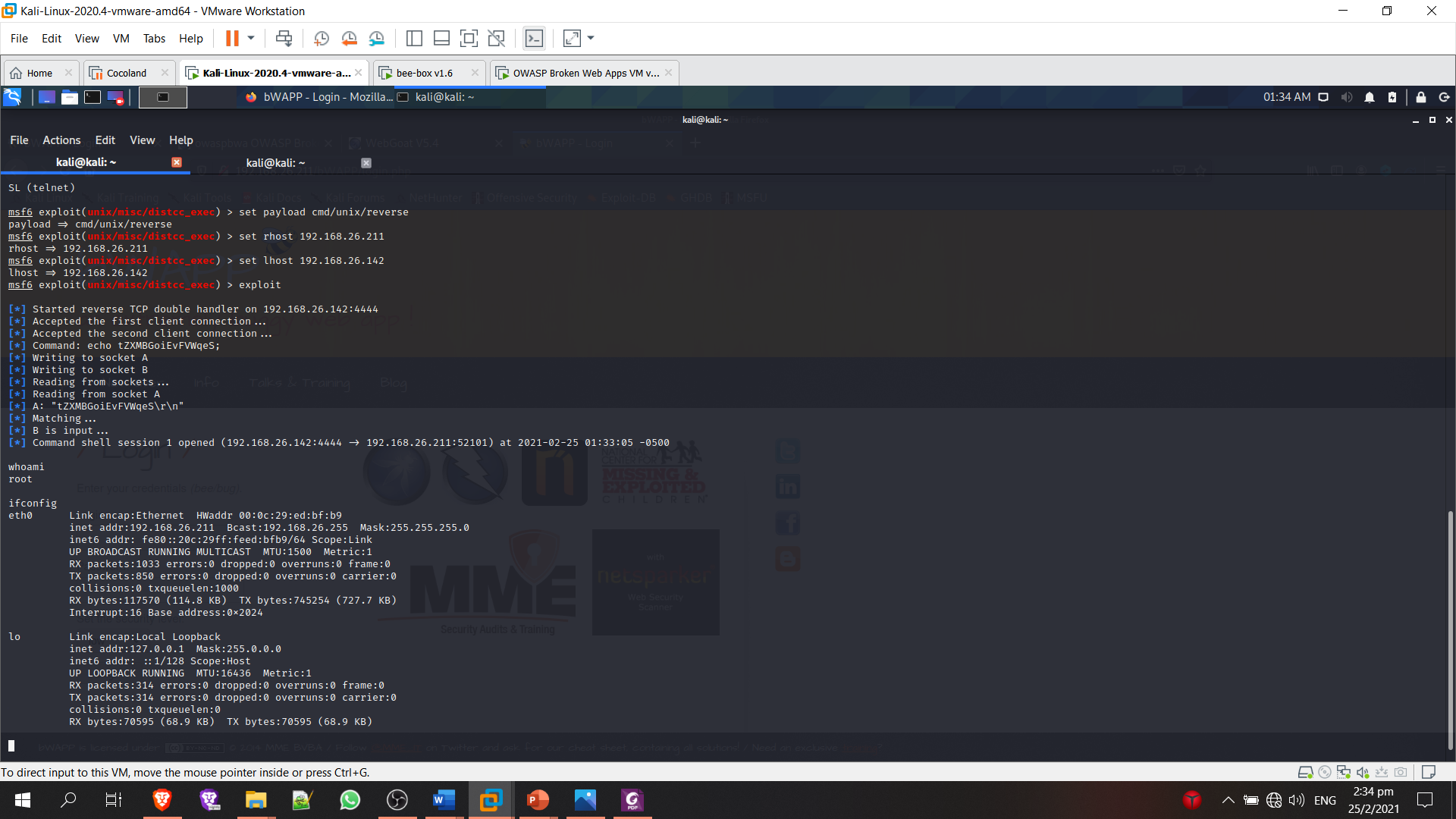
To gain shell access, I used the distcc\_exec exploit on Metasploit with the reverse payload to perform an attack through Arbitrary File Access (SAMBA) to gain access to the bWAPP server.

(10 marks)

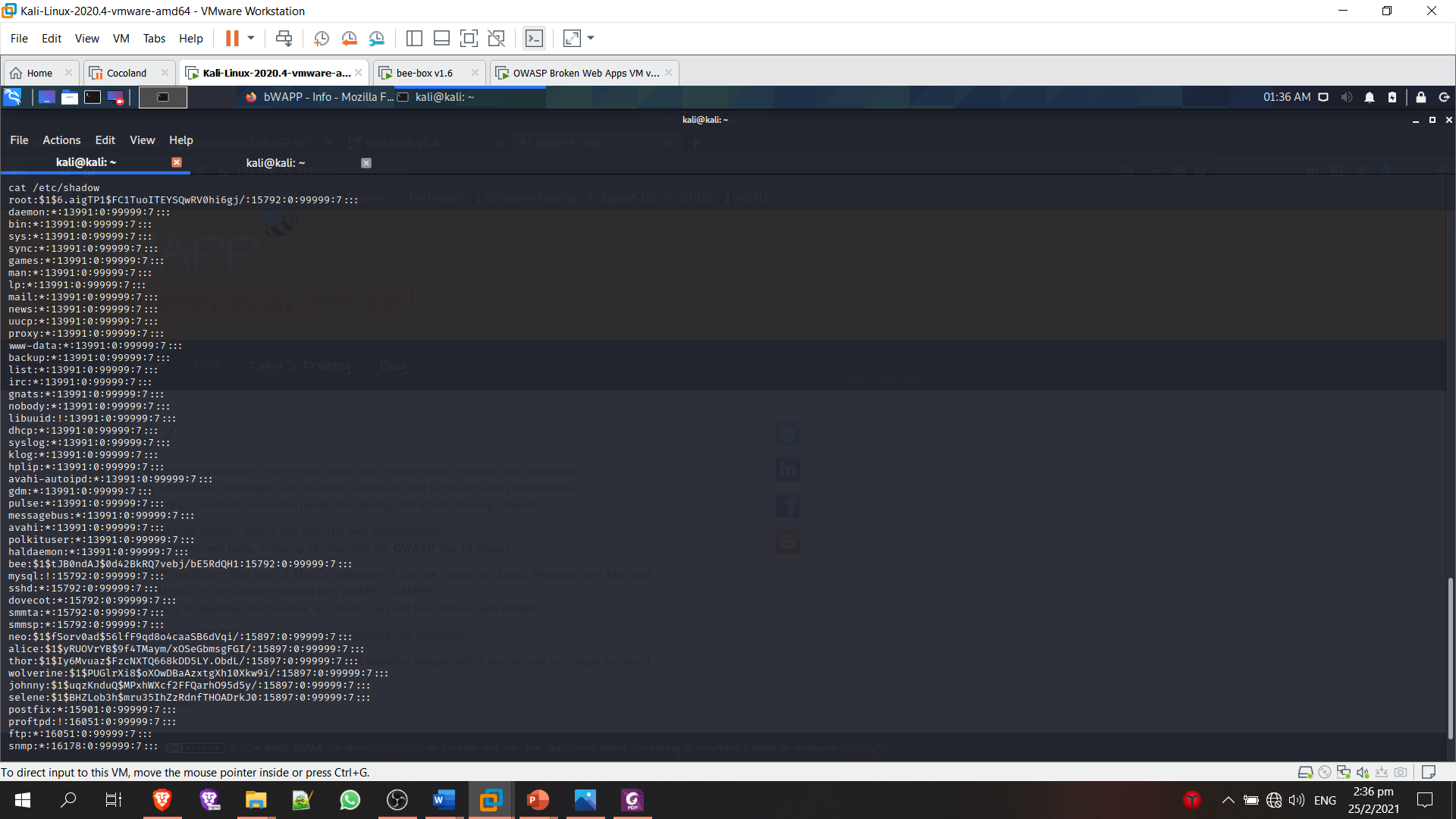
1. The post-exploitation tasks must involve the following:
2. Display the effective username under which you managed to gain the command shell session.



1. Display the configuration of the network interfaces for the server hosting the bWAPP.



1. Display the text file that contains information about the server's (hosting the bWAPP) users' passwords.



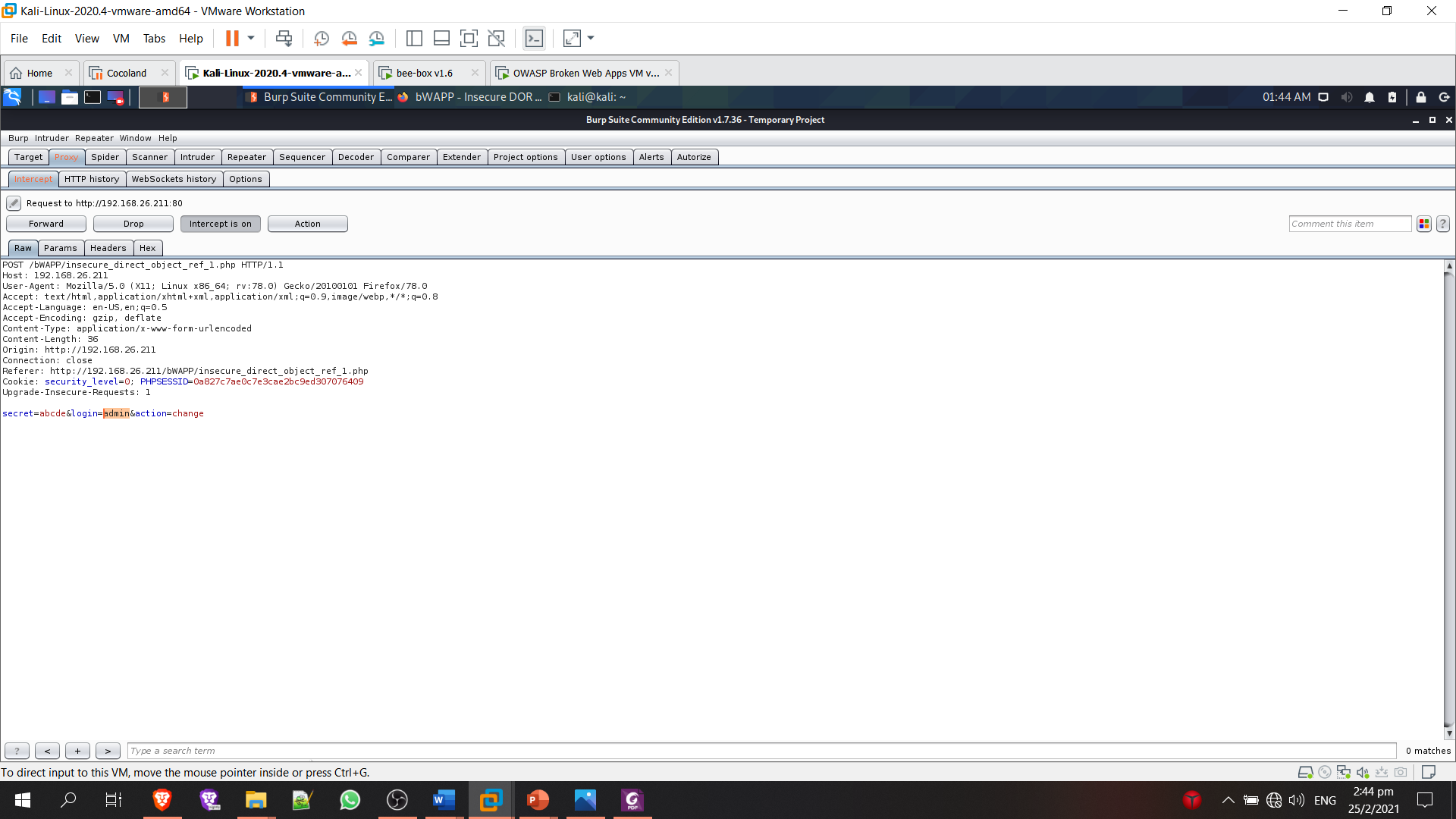
Provide screenshot(s) indicating the completion of the above tasks. The screenshot(s) must show the Host system’s (i.e., the system on which you are running your VMs) current Date and Time.

NOTE: ONLY in case you are not able to perform **QUESTION 2 (**a**)**, do the above post-exploitation tasks directly on the server hosting the bWAPP and provide the screenshot(s).

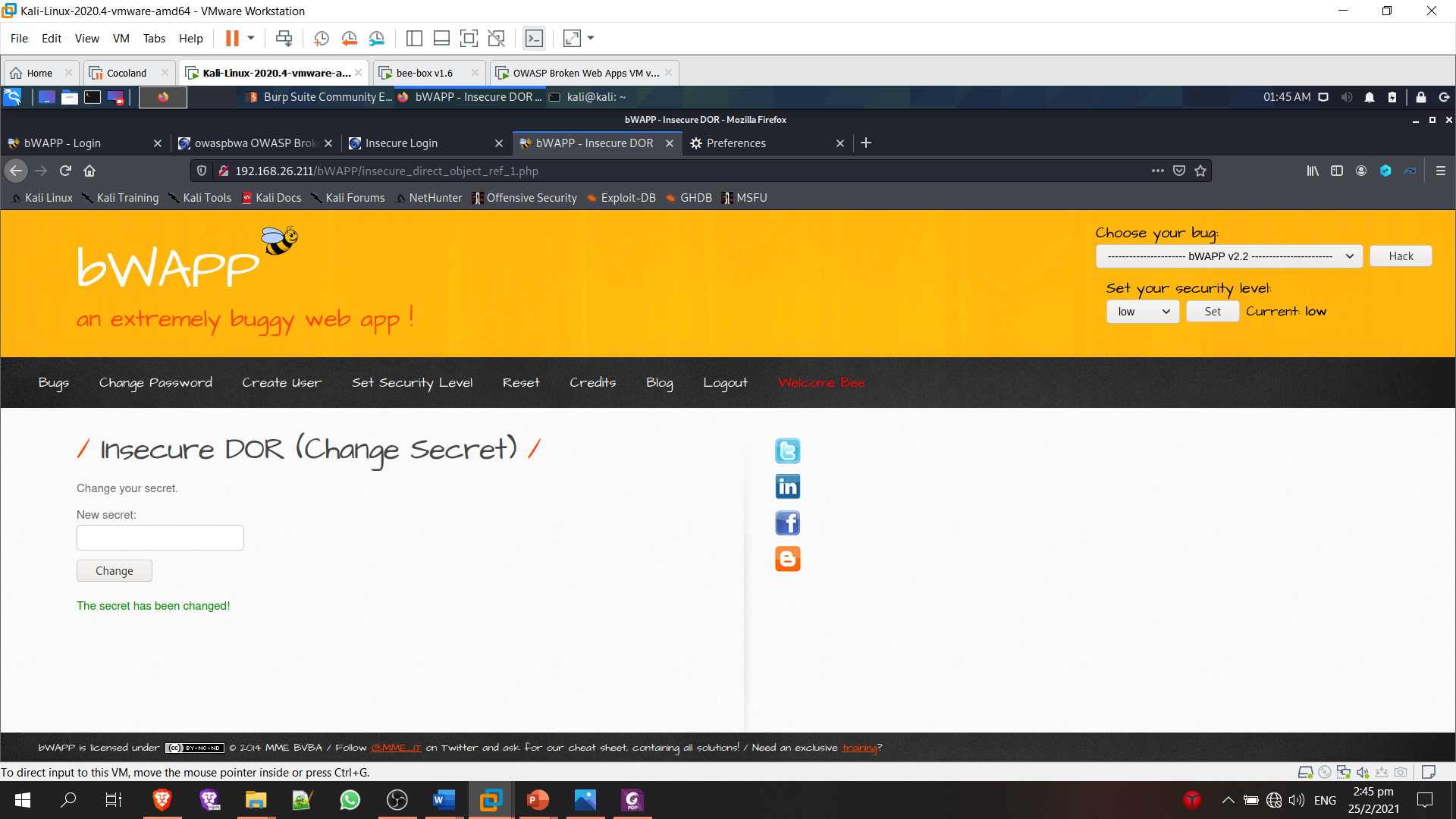
(6 marks)

1. **QUESTION 3 (18 marks)**
2. Use appropriate tools to exploit IDOR in **either** the: Owaspbwa - OWASP Broken Web Applications Project or bWAPP, an extremely buggy web app!
3. Provide detailed screenshots of the most critical steps that you took to achieve this goal, with a brief description. The screenshots must show the Host system’s (i.e., the system on which you are running your VMs) current Date and Time.

To perform IDOR exploit, I used burp suite to intercept the packet that is sent to the bWAPP server when trying to change a new secret.



I change the user from bee to admin in the packet than forwarded the packet.



The packet is sent to the server using another user account and not the attacker, this mean that we have change to secret for the user admin when we do not have the permission to do so.

(6 marks)

1. Briefly explain why the above exploit you had attempted is an IDOR?

IDOR is an attack where mostly known for privilege escalation when an application uses user-supplied input to access objects directly. In the above exploit, we sent a packet to the server where the attacker is use another user account which he/her privilege is different from the attacker.

(3 marks)

1. A web application pen-tester entered the following code:

**<?xml version="1.0" encoding="utf-8"?>**

**<!DOCTYPE foo [**

**<!ELEMENT foo ANY>**

**<!ENTITY xxe SYSTEM "http://internal-system.example.com/">**

**]>**

**<foo>&xxe;</foo>**

1. Identify which specific vulnerability is being tested here.

The vulnerability being tested are server-side request forgery which is a type of XXE injection attack.

(2 marks)

1. Analyze the above code and provide a detailed description of what it does.

The code is using xml version 1 and trying to extract all data it can get from the server with the domain of <http://internal-system.example.com/> and outputting it to the placeholder “xxe” which is in the element “foo”.

(7 marks)