**RSA**

**Module:** Cryptography

**Module Code:** COMP09106

**Activity No.:** LAB/09106/2425T1-W5

**Week:** 05

**Topic:** OpenSSL and RSA

**Duration:** Regular Class Time

**Instructor:** Dr. Althaff Mohideen

**Submission:** via Aula

**Assessment:** Summative (5%)

***Important:*** *These labs are designed to be completed on lab or home desktop/laptop environments.*

**\*\* Please do report any errata and spelling mistakes in the lab sheet if did find them.**

**Student Name:**

**Student ID:**

**After completing this lab, you will be able to:**

• Successfully understand the use of OpenSSL and RsA

**Task 01: Generate an RSA Private Key**

i. Open your terminal.

ii. Run the following command to generate a 2048-bit RSA private key:

openssl genpkey -algorithm RSA -out private\_key.pem - pkeyopt rsa\_keygen\_bits:2048

iii. The private\_key.pem file now contains your RSA private key. This file should be kept safe and not shared with anyone.

***Checkpoint 01:***

|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** | openssl genpkey: Generates a private key using the OpenSSL toolkit.  -algorithm RSA: Specifies that the RSA algorithm will be used for key generation.  -pkeyopt rsa\_keygen\_bits:2048: Sets the key size to 2048 bits for stronger security. |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* | |

**Task 02: Extract the Public Key**

i. Run the following command to extract the public key, share with your friend and receive his/her public key:

openssl rsa -pubout -in private\_key.pem -out public\_key.pem

***Checkpoint 02:***

|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** | **Explanation:** openssl rsa: Handles RSA key operations. -pubout: Extracts the public key from the private key. -in private\_key.pem: Specifies the private key to read. -out public\_key.pem: Saves the public key into public\_key.pem. |

|  |
| --- |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* |

**Task 03: Encrypt Data Using the Public Key**

i. Create a text file called message.txt containing your message: echo "This is a secret message" > message.txt ii. Use the public key to encrypt the file:

openssl rsautl -encrypt -inkey public\_key.pem -pubin -in message.txt -out encrypted\_message.bin

iii. The encrypted message is now saved in

encrypted\_message.bin.

***Checkpoint 03: Provide screenshot of the above***

|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** | **Explanation:**  -openssl rsautl: Performs RSA encryption and decryption.  -encrypt: Encrypts the input data.  -inkey public\_key.pem: Specifies the public key file for encryption.  -pubin: Indicates that the input key is a public key.  -in message.txt: Specifies the input file to be encrypted.  -out encrypted\_message.bin: Specifies the output file for the encrypted data. |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* | |

**Task 04: Decrypt Data Using the Private Key**

Now that the message is encrypted, you can use the private key to decrypt it.

i. Run the following command to decrypt the encrypted message: openssl rsautl -decrypt -inkey private\_key.pem -in encrypted\_message.bin -out decrypted\_message.txt

ii. Open the decrypted\_message.txt file to see the original message: cat decrypted\_message.txt

***Checkpoint 04: Provide screenshot.***

|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** | **Explanation:**  openssl rsautl: This command is used for RSA encryption and decryption operations.  -decrypt: Decrypts the input data.  -inkey private\_key.pem: Specifies the private key file for decryption.  -in encrypted\_message.bin: Indicates the input file containing the encrypted data.  -out decrypted\_message.txt: Specifies the output file for the decrypted message. |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* | |

**Task 05: Sign and Verify a Message**

The RSA algorithm can also be used to sign a message and verify the signature, ensuring the authenticity and integrity of the message.

i. Use the private key to sign the message.txt file:

openssl dgst -sha256 -sign private\_key.pem -out message\_signature.bin message.txt

ii. Use the public key to verify the signature:

openssl dgst -sha256 -verify public\_key.pem - signature message\_signature.bin message.txt

If the verification is successful, the command will return Verified OK.

***Checkpoint 05: Provide screenshot of the above successful CSR***

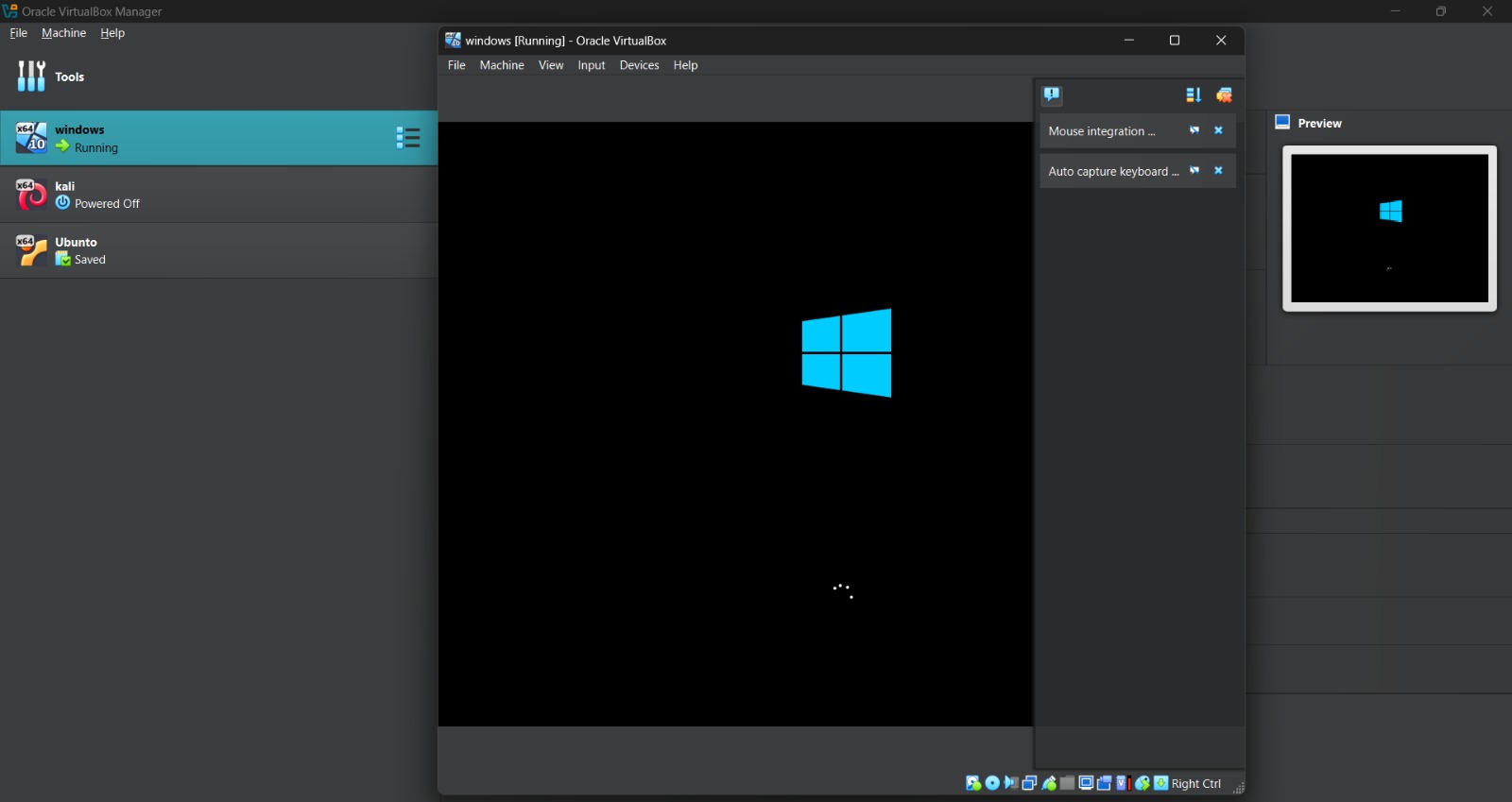
|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** | **Explanation:**  openssl dgst: Computes a message digest (hash) for files and can also be used for signing and verifying.  -sha256: Specifies the SHA-256 hashing algorithm for generating the digest.  -sign private\_key.pem: Uses the private key to sign the message, creating a signature.  -verify public\_key.pem: Uses the public key to verify the signature of the message.  -signature message\_signature.bin: Indicates the file containing the signature to be verified.  message.txt: Specifies the original message file that was signed. |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* | |

**Task 06: Challenge Task**

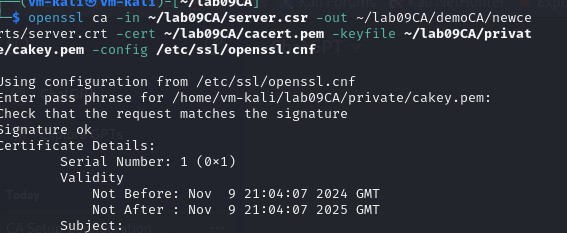
***Use your friend’s public key to encrypt a message and email it to your friend. Request your friend to decrypt the message, take a screenshot and send it to you.***

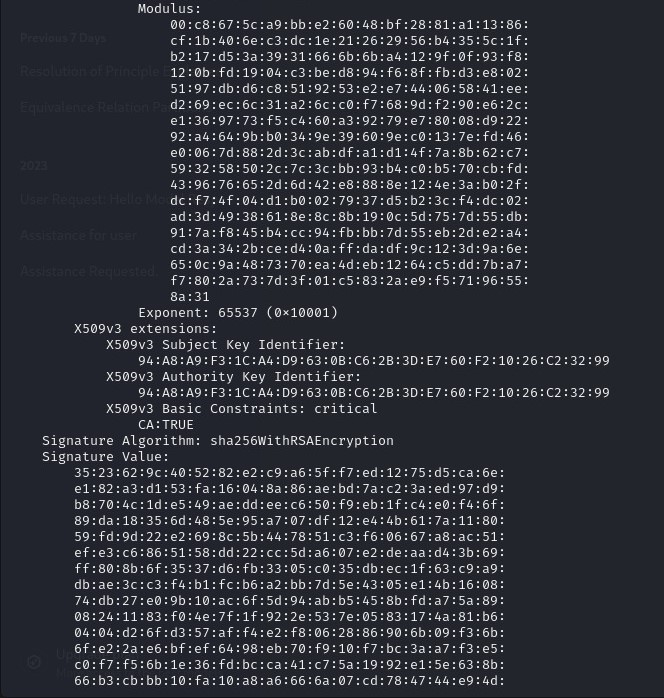
|  |  |
| --- | --- |
| ***Date:*** |  |
| ***Evidence/Snapshot:*** *(Checkpoint)* |  |
| ***Description/Detail:*** |  |
| *Note: Copy/Paste this table to provide multiple evidence or snapshots.* | |

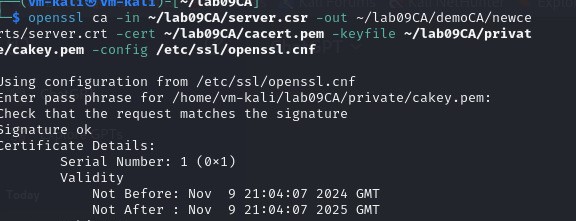
***Here is the below some command that we have run on kali and ubuntu on the virtual box ---***

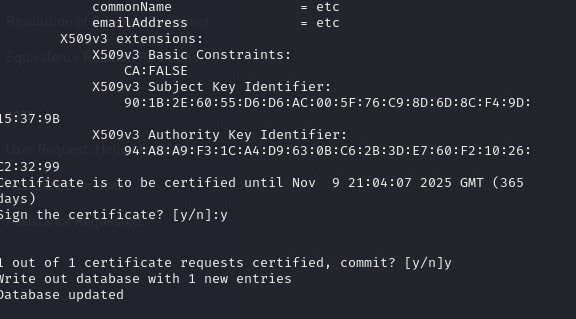
******

***CA(VM) certification commands :***

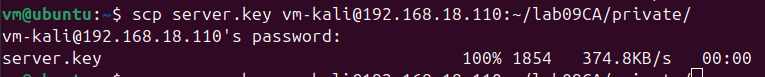
******

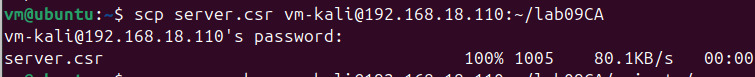
******

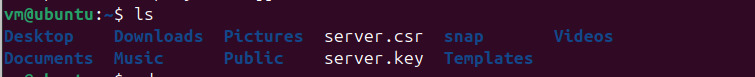
******

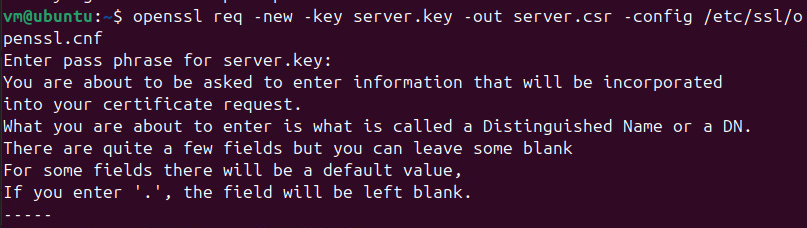
******

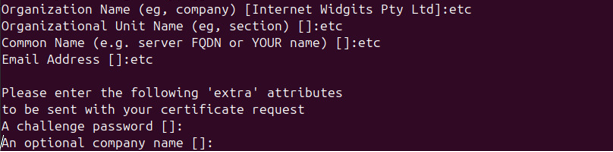
***Commands that we have on vm ubuntu tranfer file using (SCP):***

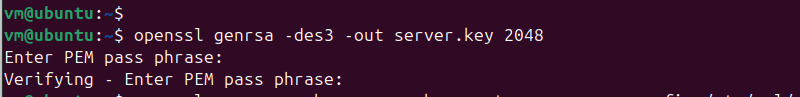
******

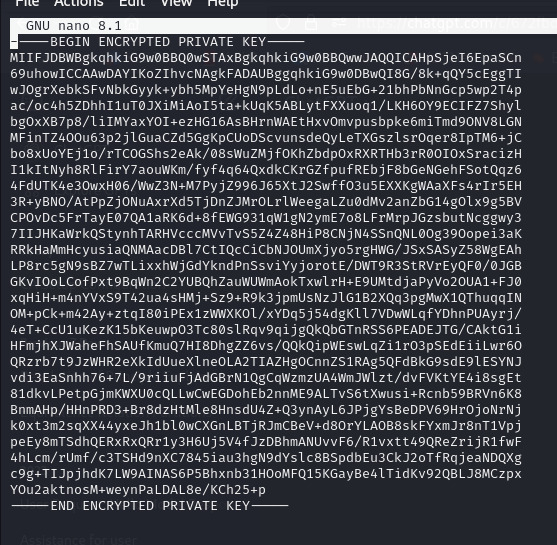
******

******

******

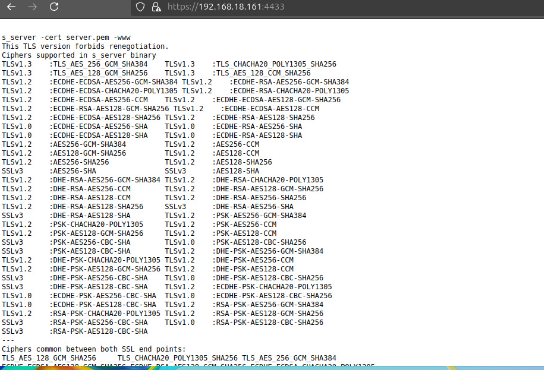
******

******

******

***Here we use HTTPs Apache2 for showing the secure connection on the web brower of VM ubuntu :***

***In the case https://vm-kali-ip:4433***

******