Table of Contents: **Page**

Introduction

This coursework simulates the tasks of an ethical hacker who is responsible on finding and fixing the vulnerabilities in a hypothetical company's network. This network is designed with a number of security flaws, and they are of three main types: network design, encryption methods, and authentication procedures. These flaws were discovered using penetration testing, which is a method of evaluating the security of a computer network by simulating an attack.

For this coursework is required:

- Access to a laboratory that is strictly controlled and secure
- Endpoint Security VPN, a virtual private network
- Investigative Kali Linux for penetration testing
- Remote Access via VMware Horizon Client
- Tools for cryptographic analysis, such as SageMath

Steps to be done:

- Gaining access to the investigator machine
- Using a dictionary attack to exploit SSH admin credentials
- Retrieving the SQL credentials from the encrypted email
- Finding the sensitive database information by accessing to the SQL server
- Using various cryptographic techniques to recover credit card secret code

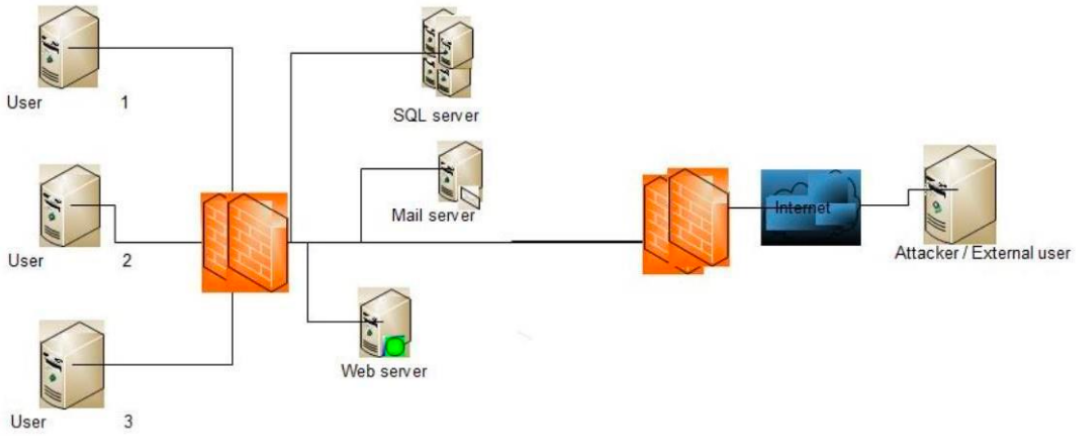
This coursework highlights common security vulnerabilities. These are result of weak key management, poor encryption setups, and insecure password schemes. It uses tools like Nmap, Hydra, and SageMath, to show just how much ethical hacking can help find and fix these problems. In a nutshell, finding those problems and fixing them makes our cybersecurity frameworks much more secure.

Work Plan

System Overview:

CyberSec's platform provides a safe way to conduct online transactions and protect the private information of clients. The system is carefully architected to maintain and protect service availability, confidentiality, and integrity. It has several layers of defence and a number of important components. Here's an outline of some of the most important ones:

Network/Map Diagram:



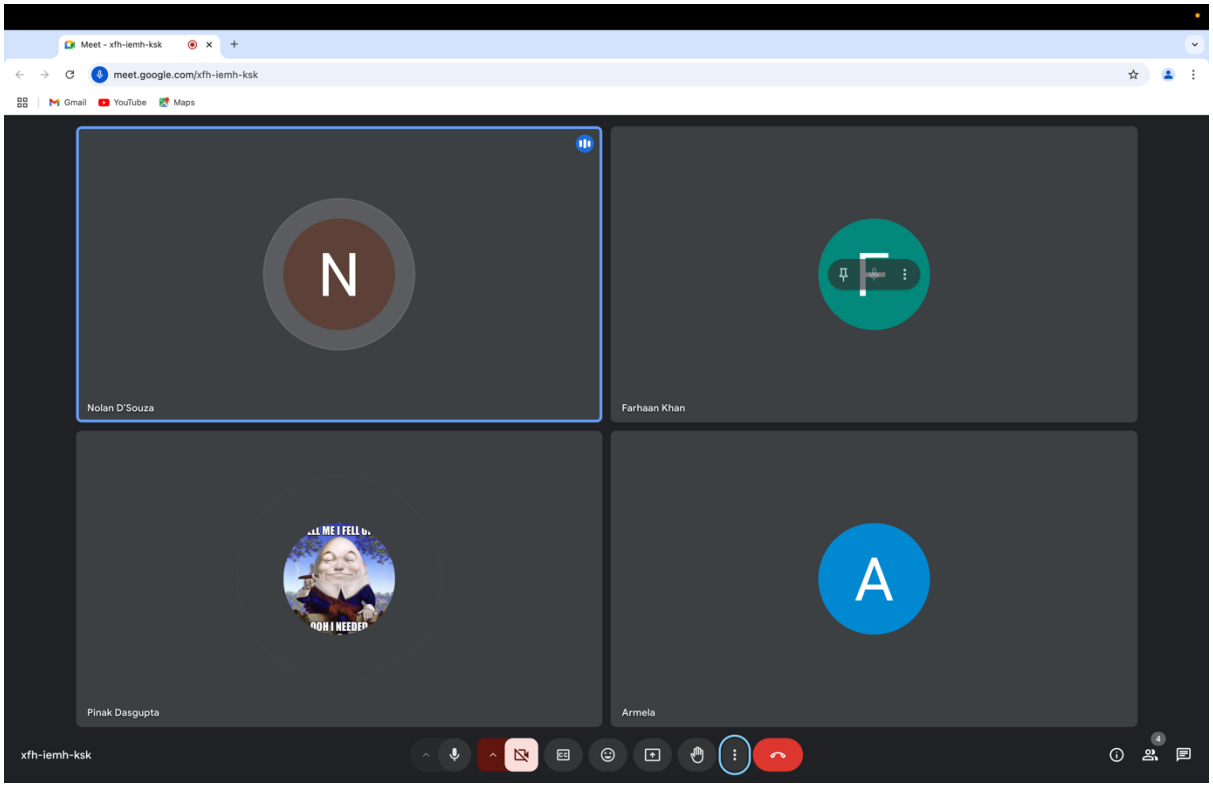
1. Firewall
   All incoming and outgoing traffic is controlled by firewall, which is the outermost layer. When think of firewall, you should think of one at the network level. This drives into your head that a firewall exists to protect access to something on a network or someone's else's.
2. DMZ
   The DMZ serves as a buffer zone between the internal network and the external one. It is where the essential services needed to conduct business and make profitable cyber selection operate:
   - Web Service provides the online product browsing.
   - Mail Service: Oversees all correspondence and the sharing of documents and records.
   - SQL Database
3. SQL Database: Contains client's confidential information.

4. The inner firewall: Is a safeguard that further secures an company's internal network, even beyond what is already offered through the external firewall.

Team Collaboration

Together as a group, we gave each other a set of responsibilities, including things like reconnaissance, credential exploitation, and data extraction. Using the Extended Euclidean algorithm was the most important part of the coursework, where we used Shamir's Secret Sharing Scheme with the information from the working group table that was given. Everyone had to finish their own part of the calculations, which was checked by all the team members afterwards.

Frequent gatherings, mostly via Google Meet with a few in-person meetings helped us to talk about each group member progress and improve the calculations.



The first meeting that took part was about reconnaissance:

- The goal of this meeting was to comprehend the layout and configuration of the network.
- To find hosts and open ports by using tools like Nmap and ifconfig.

The second meeting that took part was about Brute Force and Email Decryption:

- Here it was mostly discussed about locating the SSH server for the brute force that was going to be used.
- Also, about the email that had to decrypt manually by everyone.

The third meeting that took part was about Shamir's Scheme Calculations:

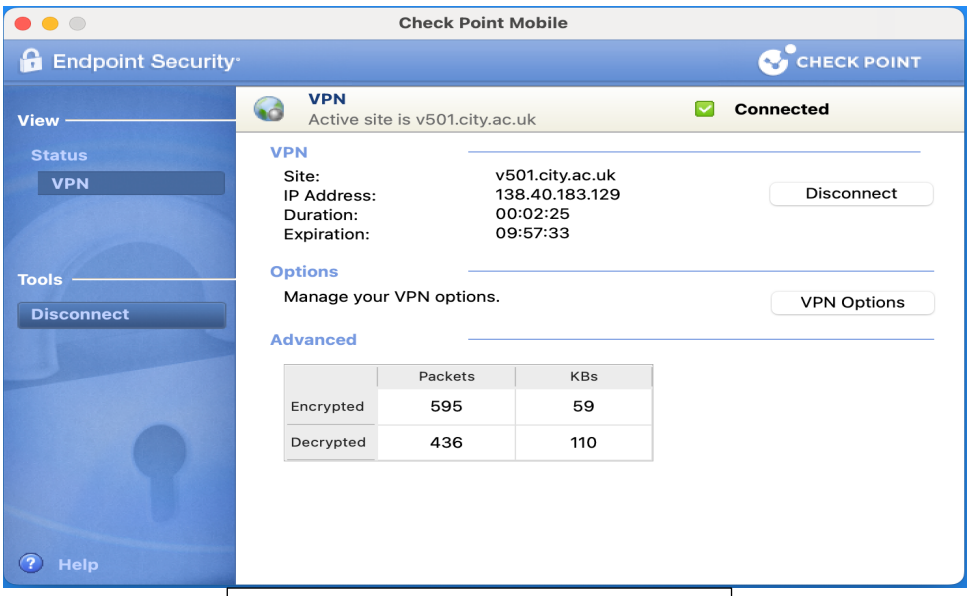
- We did this meeting to discuss about the calculations that we needed to do by hand, where each of the team members had to calculate their part.
- We shared ideas on how to calculate this part of the coursework to give us accuracy and efficiency in the methodology used.

The fourth meeting that took part was about calculating the Public Key *e*:

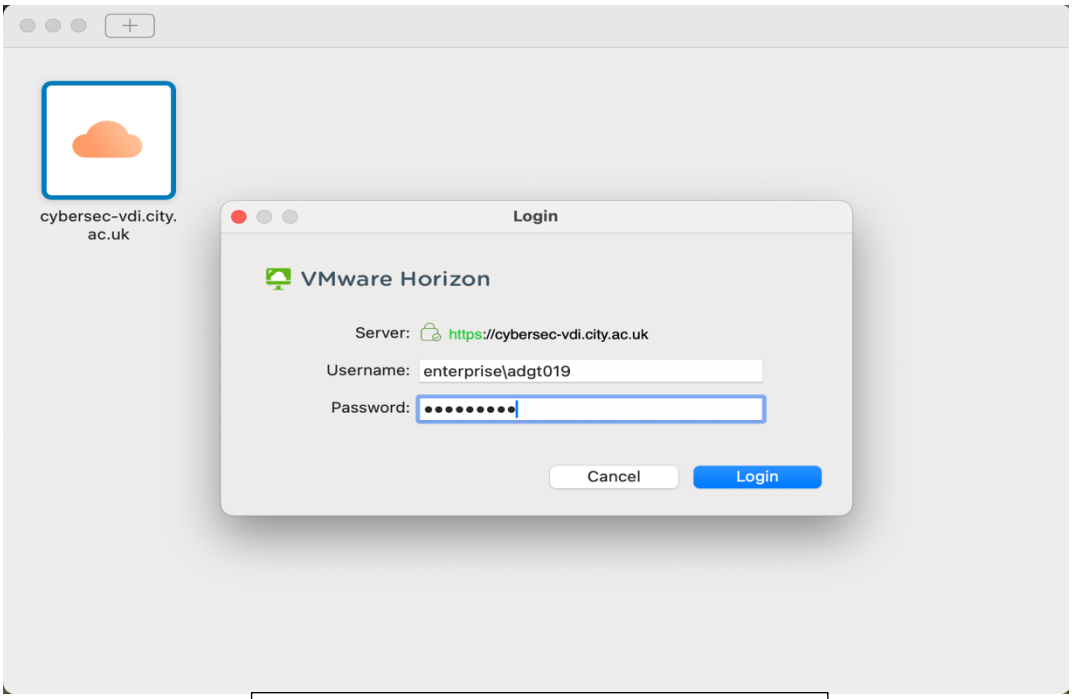
- In this meeting we shared with each other the results that we needed for the public key calculations.
- This meeting helped us to have the results correctly done.

Getting access to the investigator

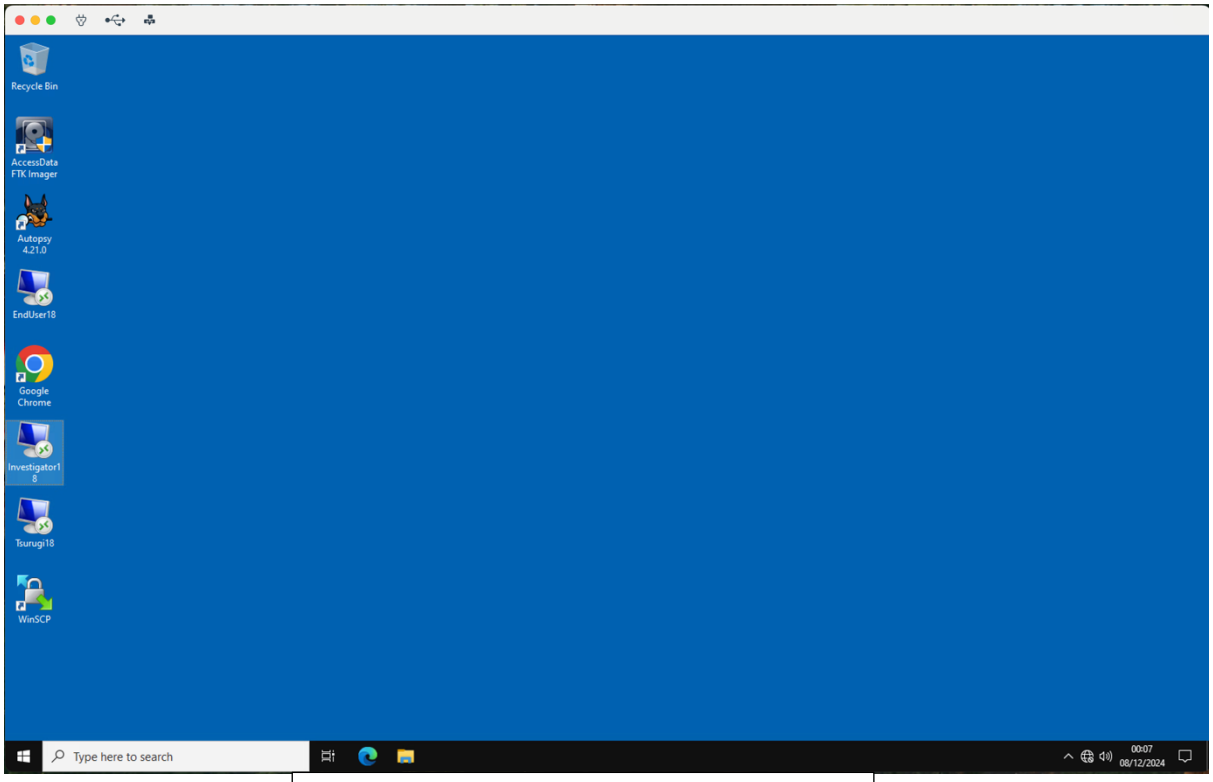
Firstly, I must access the investigator machine to start with the coursework and to finish each step. Here I will provide with some screenshots of how I accessed the investigator machine:



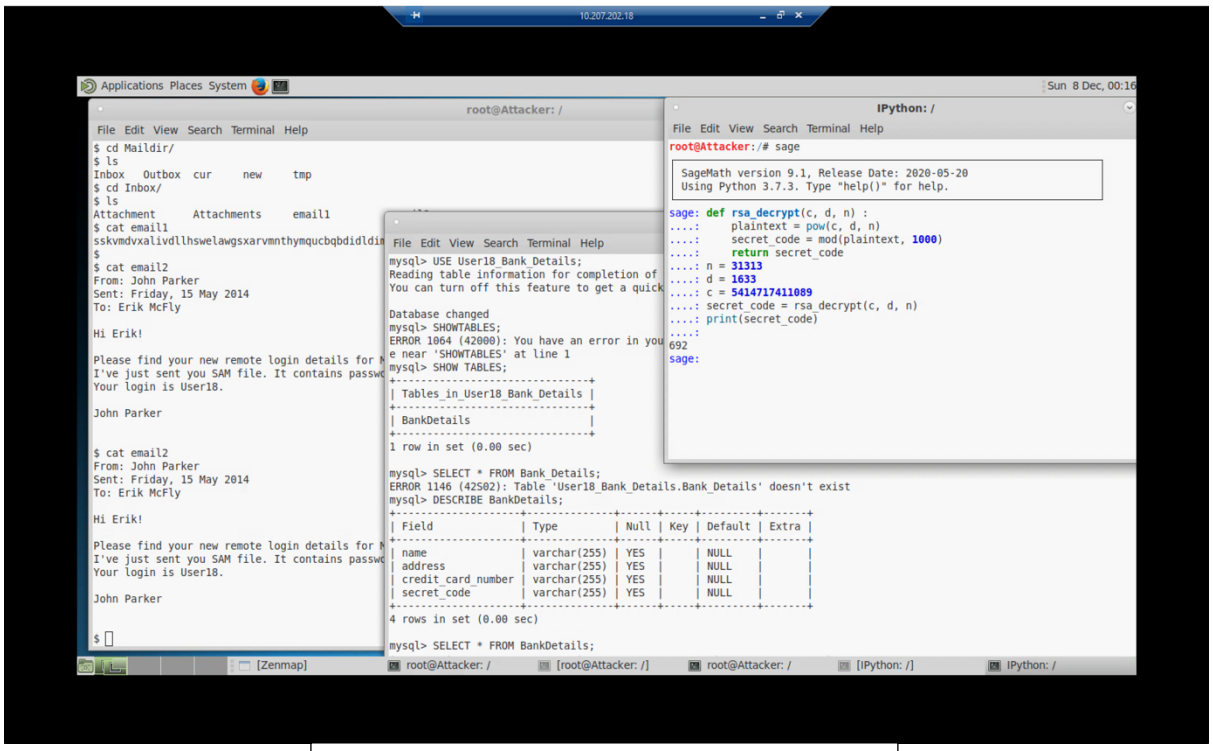
VPN connection



VMware Horizon Client Login



The Windows Machine



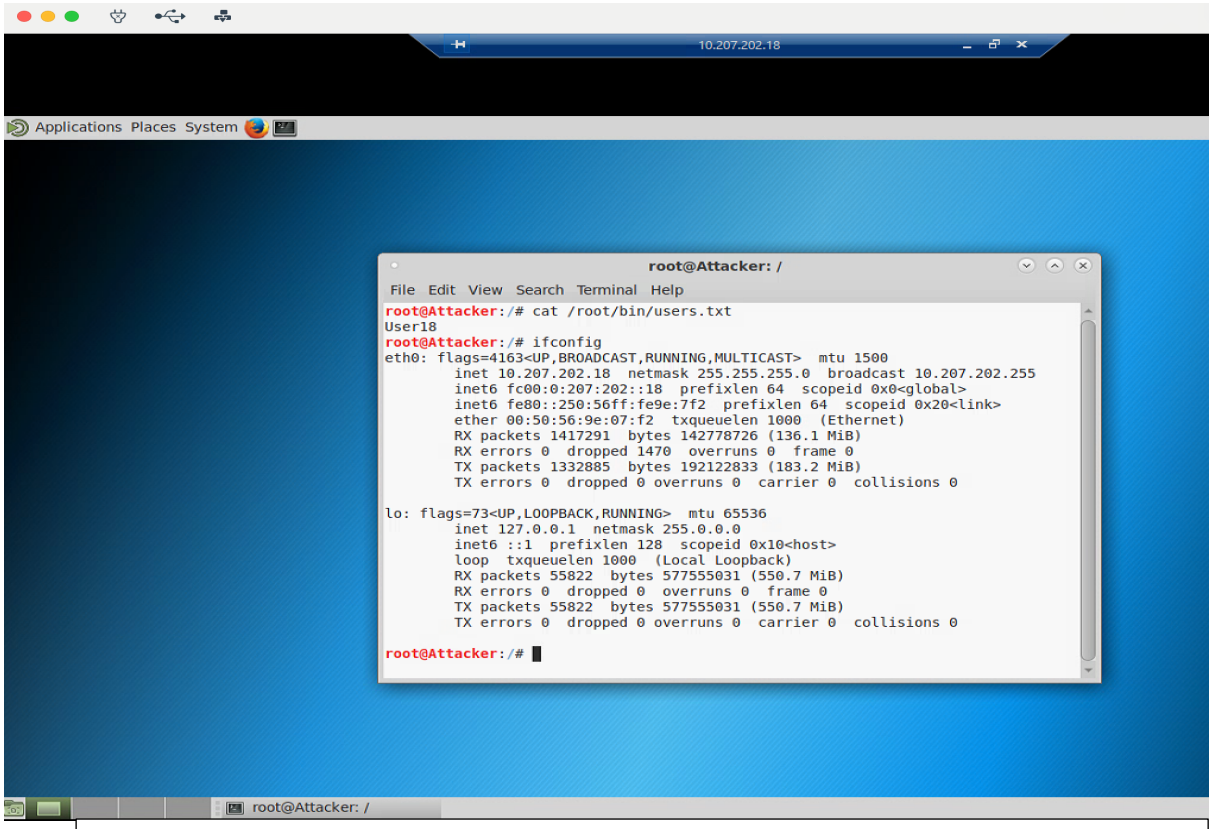
My Investigator Machine

Network reconnaissance and vulnerability

By inspecting the file /root/bin/users.txt, it was confirmed that the username in use is **User18**.

The network configuration of the attack machine was identified using the **ifconfig** command:

IP Address: The attacker machine from which the attack originates has been given the IP address **10.207.202.18**



Verification of Attacker Machine Configuration and User Details

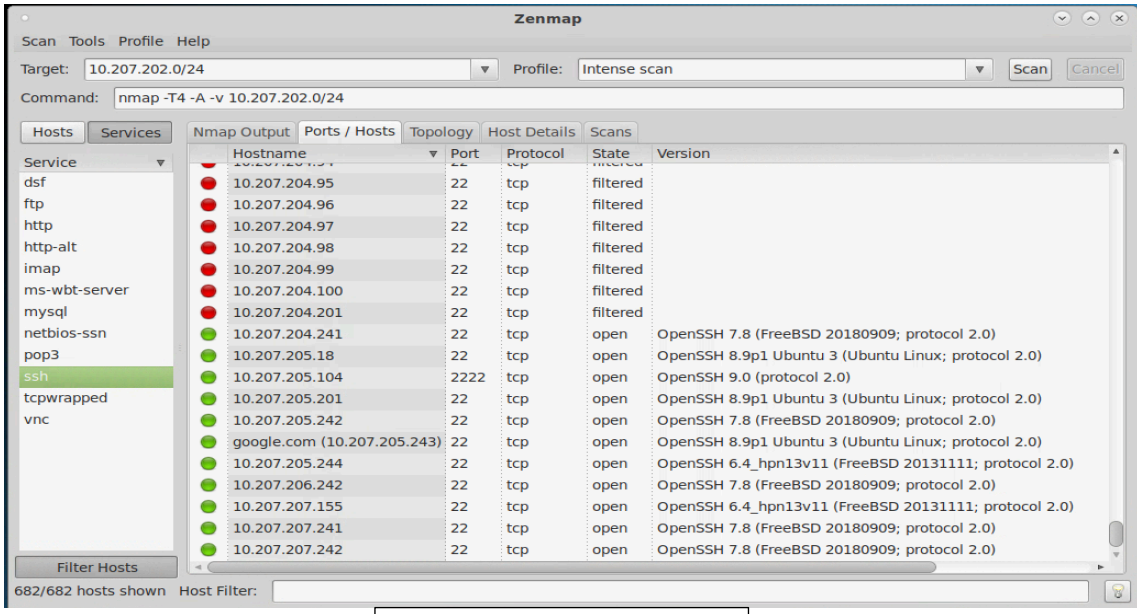
The goal of this task is to discover which hosts are active, which ports are open, and which services are running within the target network (10.207.202.0/24). Finding these things will allow the project team to determine if there are any exploitable vulnerabilities in the target network.

Methodology:

- An intensive scan was performed using a graphical Nmap interface, Zenmap with the following command: nmap -T4 -A -v 10.207.202.0/24
- This scan was configured to gather detailed information about active hosts, open ports, service versions and operating system details.

Key findings:

1. Active Hosts: The subnet contained many hosts that were active and had open or filtered ports.
2. Open Ports and Services: Among the various hosts, we noted that port 22 (SSH) was open on several of them. An attacker could try to exploit this port by cracking any weak passwords that might be associated with its service.
3. Filtered Ports: A few of the hosts' ports seem to be filtered, probably via a firewall, which reduces the attack surface.
4. This showed that SSH configurations might have vulnerabilities, especially if they had weak passwords.



Zenmap Scan

Identifying potential attack surfaces depends heavily on using Zenmap/Nmap. These tools provide the detailed output necessary for ethical hackers to zero in on targets.

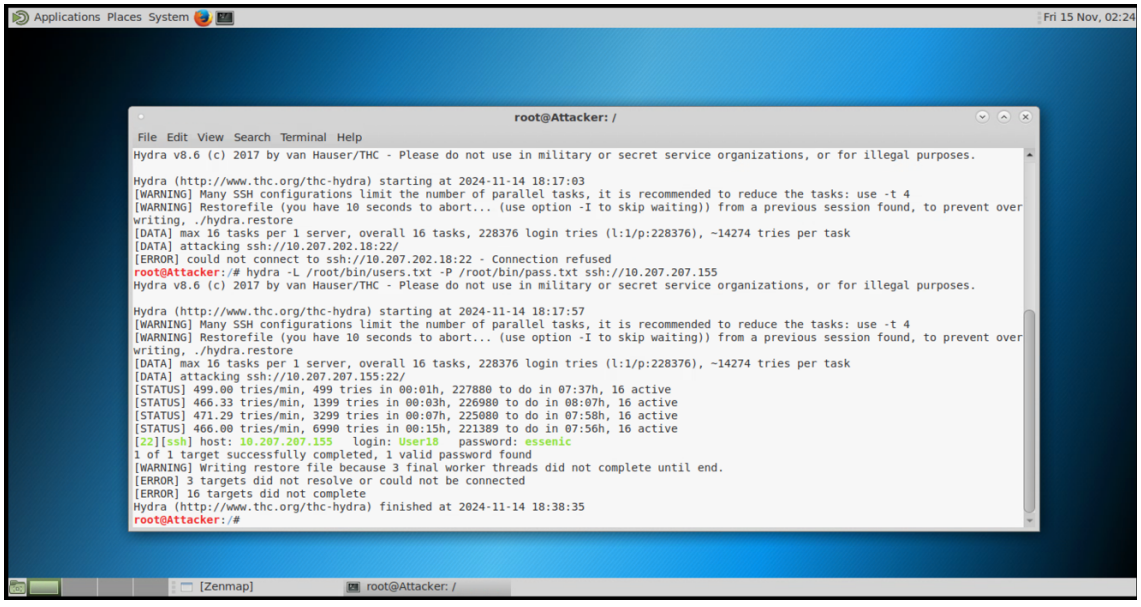
Performing a dictionary attack in the admin password to gain access to the SSH server:

Objective: To carry out an unauthorized SSH login by exploiting a weak password for the root account. Activity demonstrates the risk of using easily guessable passwords and simulates a controlled, real-world scenario in which attackers could use a list of common passwords to gain unauthorized access to a server.

Methodology:

- The tool used was Hydra, a speedy and efficient password-cracking tool. It is well-known for its support of various protocols, among them SSH.
- Issued commands:
  hydra -L /root/bin/users.txt -P /root/bin/pass.txt ssh://10.207.207.155
    - -L /root/bin/users.txt: Identifies a file that likely has usernames.
    - -P /root/bin/pass.txt: Identifies a file that likely has passwords.
    - ssh://10.207.207.155: Directs to attempt to log in via SSH (a remote login service) on the named IP address, which is 10.207.207.155.

Results:



Systematic testing of password combinations from the dictionary file was carried out by Hydra. This versatile tool was employed because of its capacity for efficient handling large files of passwords and, of course, its deftness with the SSH service.

Real-World Considerations:

- Managing Large Datasets: Divide dictionaries into smaller segments using the split command to enable more manageable processing. Prior to use, sort lists of passwords based on their degree of entropy, giving clear priority to those that are common and most likely to used.
- Attacks from many parts: Used the components of a distributed system in parallel to carry out an attack and tools like Hydra clusters to make the attack scale better.

Other tools that could be used:

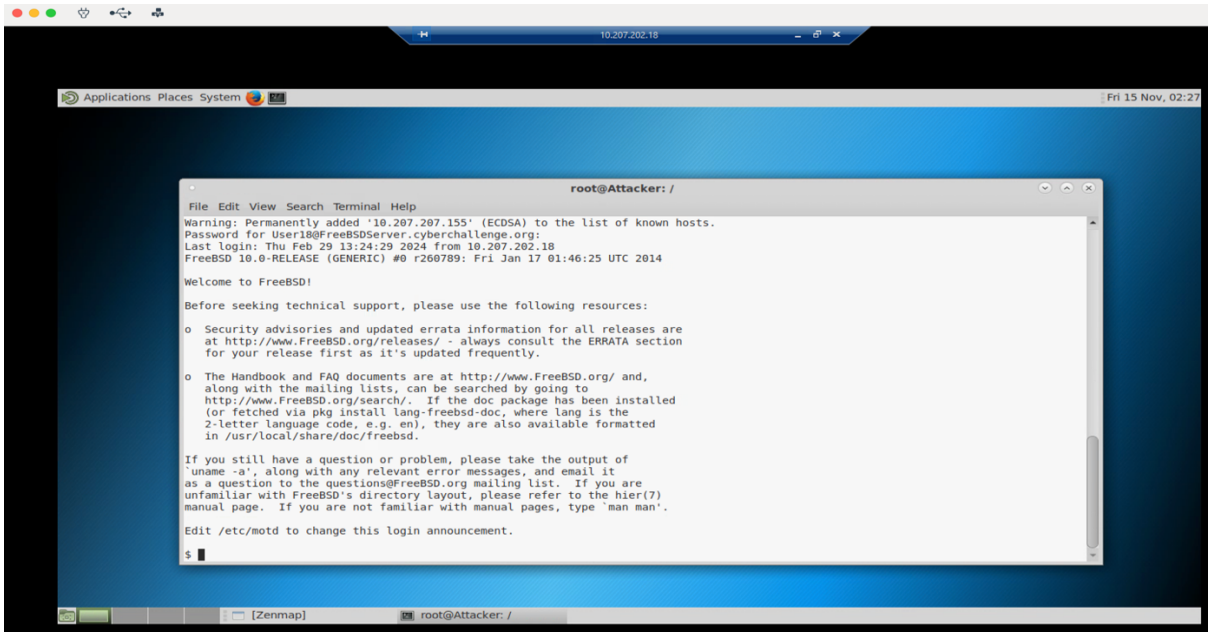
- Medusa: Known for its straightforwardness and effectiveness.
- Ncrack: Concentrates on breaking authentication systems targeted at networks.
- These tools can handle specific scenarios, like working with huge datasets or uniting with specialized workflows.

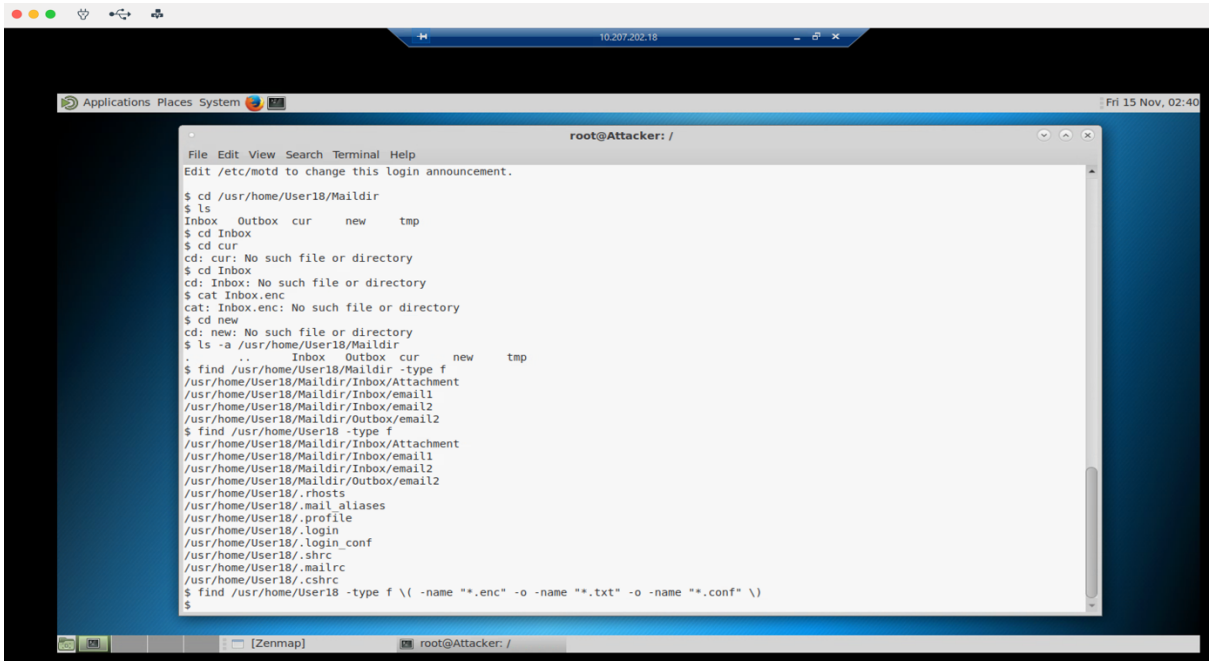
Key Results: The password essenic, which was discover using Hydra tool led to a successful login.

Objective: The next step after acquiring SSH access was to find and examine user data, particularly email data for sensitive information.

Process:

- I connected to the server at 10.207.207.155 with the credentials I acquired earlier (username: root, password: essenic).
- The emails and their attachments were located through the directory /usr/home/User18/Maildir.





Email Analysis

Results from the Maildir: Located emails in the /Inbox and /Outbox directories. These contained sensitive information, which involved user IDs and these kind of private data that would allow hackers to compromise a server.

The details from the email are as follows:

Email 1:

sskvmdvxalivdllhswelawgsxarvmnthymqucbqbdidldimyyuggkwhribmwnqy
qsdborwhobisdbolxcrbighsfiqimdlutskrgfcxikjbnvhssskkimyrhgsqeusgcqliv

Email 2 (Outbox and Inbox emails):

To: Erik McFly

Hi Erik!

Please find your new remote login details for MySQL Server in the attachment.

'we dugt is ser sal fale. It contains password in LM-hash(DES).

John Parker

From: Bert McFly

Sent: Monday, 12 May 2014

To: John Parker

Hi John!

I've desided to connect our Snort Intrusion Detection System

to the

SQL database.

Could you send me please new login/password information for

MySQL Server access.

Cheers.

P.S.

Don't forget to encrypt your email for the security reasons.

Let's agree

on LM DES.

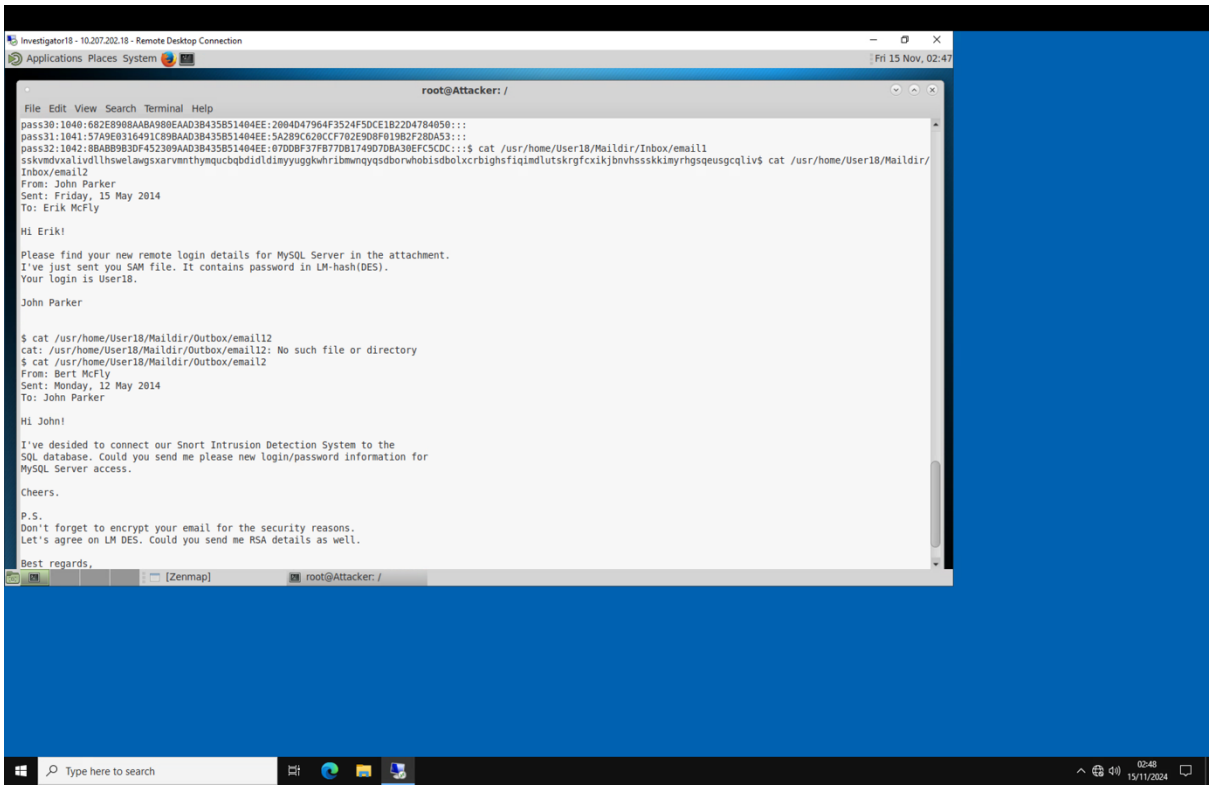
Could you send me RSA details as well.

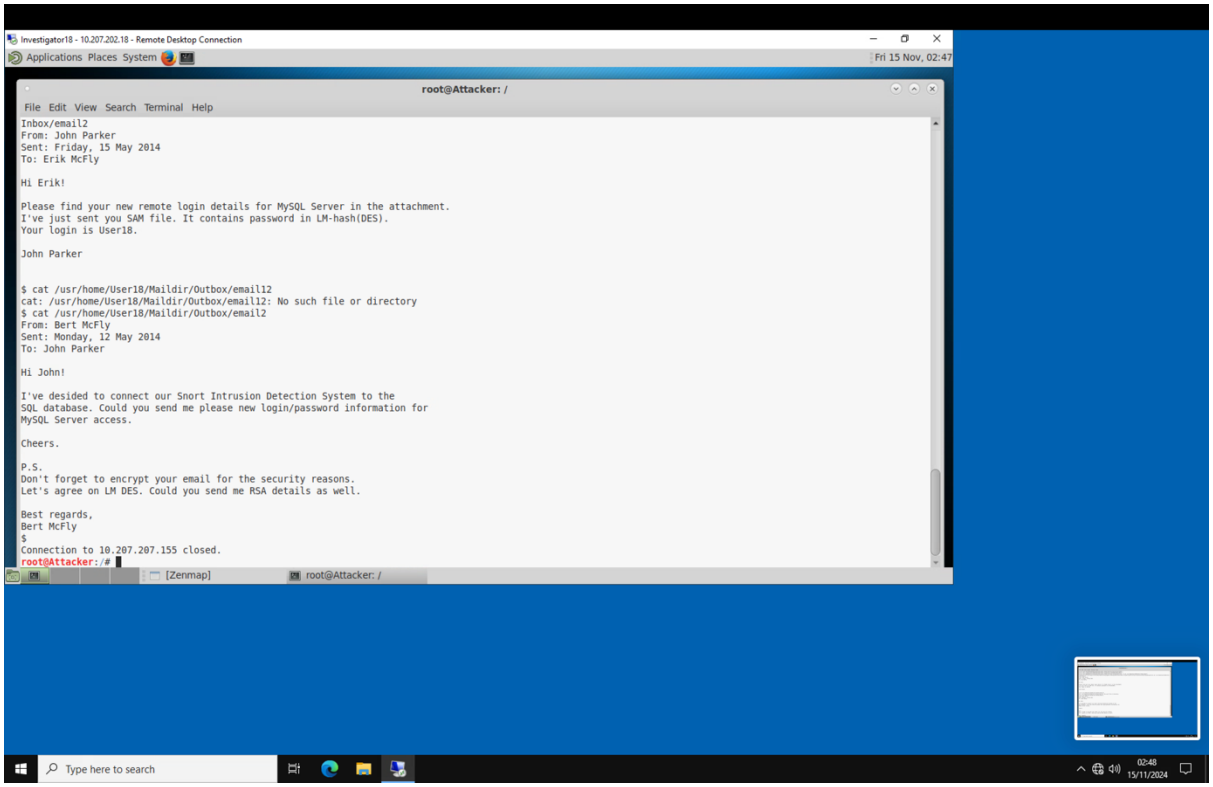
Best regards,

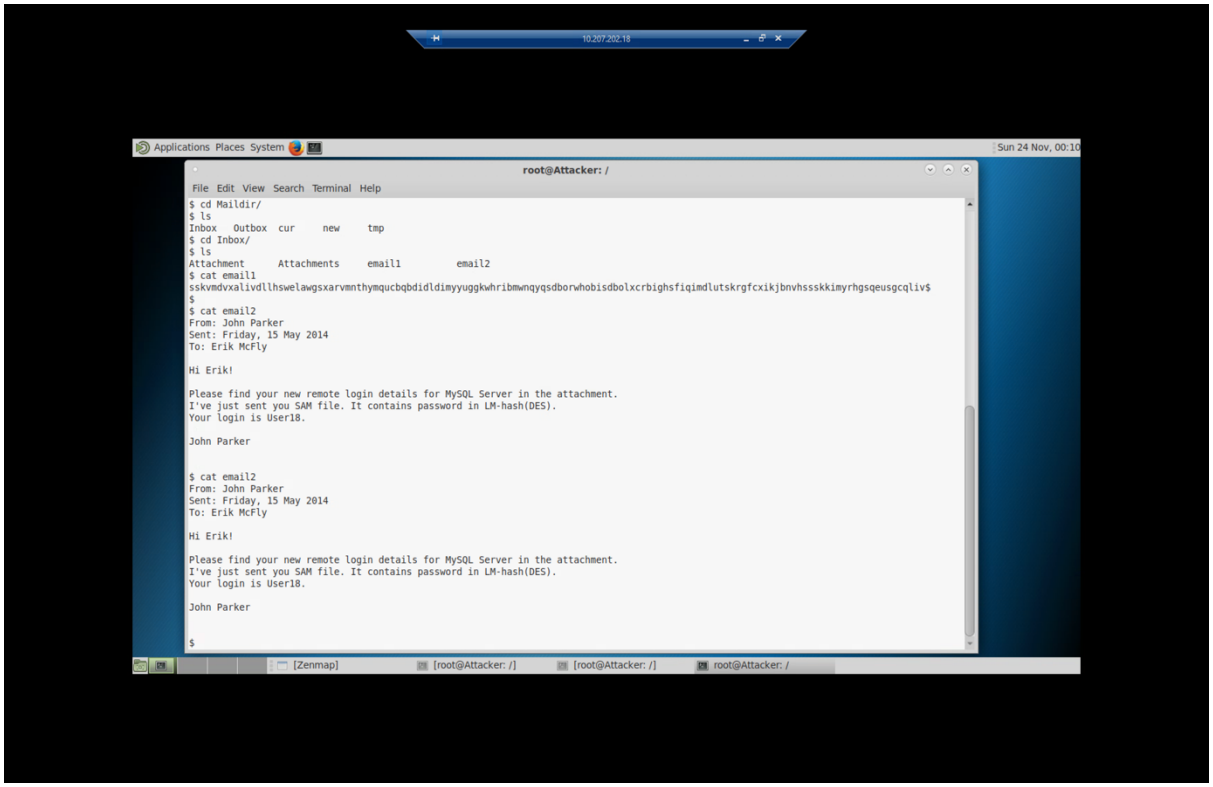
Bert McFly

```
$ cat email1
sskvmdvxalivdllhswelawgsxarvmnthymqucbqbdidldimyyuggkwhribmwnqyqsdborwhobisdbolxcrbighsfiqimdlutskrgfcxikjbnvhssskkimyrhgsqeusgcqliv$
$
```

Encrypted Email







The email provides the details for the decryption key that I will use to access encrypted files and to search for customer data.

I used basic Linux commands, like cat and find, to investigate the directory structures and retrieve the essential information efficiently.

Importance of Email Analysis: Emails frequently hold sensitive data, such as encryption keys and login credentials. If misused, this kind of data can provide a means for reaching databases or elevating one's privileges within a system.

Cryptanalysis of the encrypted email:

Objective: This work aims to uncover the hidden message carried by the ciphertext without using any internet-based tools or resources. Instead, it relies on a thorough understanding of the tools and techniques of cryptographic analysis. These techniques involve the close examination of the ciphertext with the hope of revealing some clue to its meaning. They also require a coequal examination of the possible "keys" that might have been used to encode the message and the determination of which key, if any, might be the correct one.

Methodology:

1.Analyzing Frequencies

- Aim: The main goal of frequency analysis is to recognise the characters that appear most often in the ciphertext, essentially, to find what is from an English perspective the most used letter in the code. The reason for this is obvious: Once I have identified which character corresponds to 'E', I have already made a significant stride toward decrypting the rest of the message.
- Approach: Counted how many of each character appear in the cipher text. Then determined the frequency of occurrence for each character, expressing it as a percentage of the total number of characters in the cipher. The most reliable information comes from the characters with the highest frequencies, which can then be used to make some reasonable guesses about which character they correspond to in the plaintext.

2. Identifying Repeating Patterns

- Aim: Repeating patterns in ciphertext often suggest periodicity, indicative of a polyalphabetic cipher like Vigenère. These patterns help determine the key length.
- Approach: My work involved spotting sequences that repeated in the ciphertext and figuring out where exactly they were positioned. After that, I worked out the measurements that told me how far apart these repeated sequences were. Once I had a good handle on that, I used the information to find even more patterns and to get a better idea of what the key length is.

3. Key Length Calculation:

- Aim: The length of the key is of high importance because it determines how to group the encrypted text into columns that can be decrypted one by one.
- Approach: I determined the probable key length of the cipher by calculating the distances between repeated patterns in the text and then finding the greatest common divisor of those distances. I used the well-known Euclidean Algorithm to perform these calculations, which often offers an efficient way to find the GCD.

4. Group Shift Calculation:

- Aim: The previous step established the key length. Now, I take the next logical step and divide the ciphertext into segments that correspond to the positions in the key. I analyse the segments to infer the shifts necessary for a successful decryption attempt.
- Approach: The process I followed assumed that the character most commonly occurring in each group represented the letter ‘E’ in English. From there, I computed the shift for each group and noted the shifts to decipher the key.

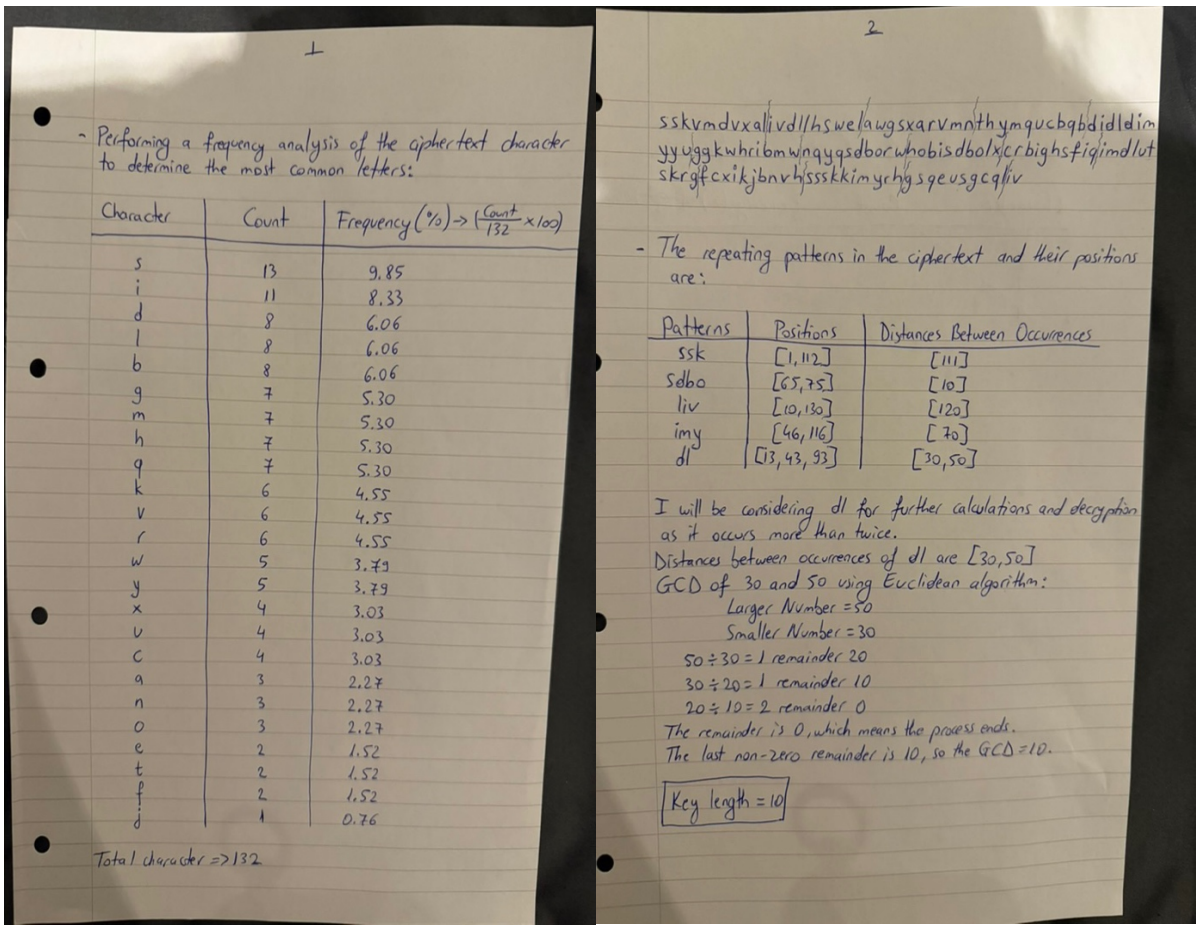
5. Decoding the Encrypted Message

- Aim: To use the derived key to perform decryption on the ciphertext to yield the original plaintext.
- Approach: To decrypt the message, first I find the corresponding shift for each character in the key. I adjusted for modular arithmetic, when

necessary, then subtracted the shift from the corresponding ciphertext characters. I repeated this for the entire ciphertext, cycling through the key as needed.

The message below was produced at the end of the calculations:

dear john with this email i would like to inform you that your rsa security components are one hundred seventy-three and one hundred eighty-one rewards smith



1

- Performing a frequency analysis of the ciphertext character to determine the most common letters:

Character	Count	Frequency (%) → ($\frac{Count}{132} \times 100$)
s	13	9.85
i	11	8.33
d	8	6.06
l	8	6.06
b	8	6.06
g	7	5.30
m	7	5.30
h	7	5.30
q	7	5.30
k	6	4.55
v	6	4.55
r	6	4.55
w	5	3.79
y	5	3.79
x	4	3.03
u	4	3.03
c	4	3.03
a	3	2.27
n	3	2.27
o	3	2.27
e	2	1.52
t	2	1.52
f	2	1.52
j	1	0.76

Total character => 132

2

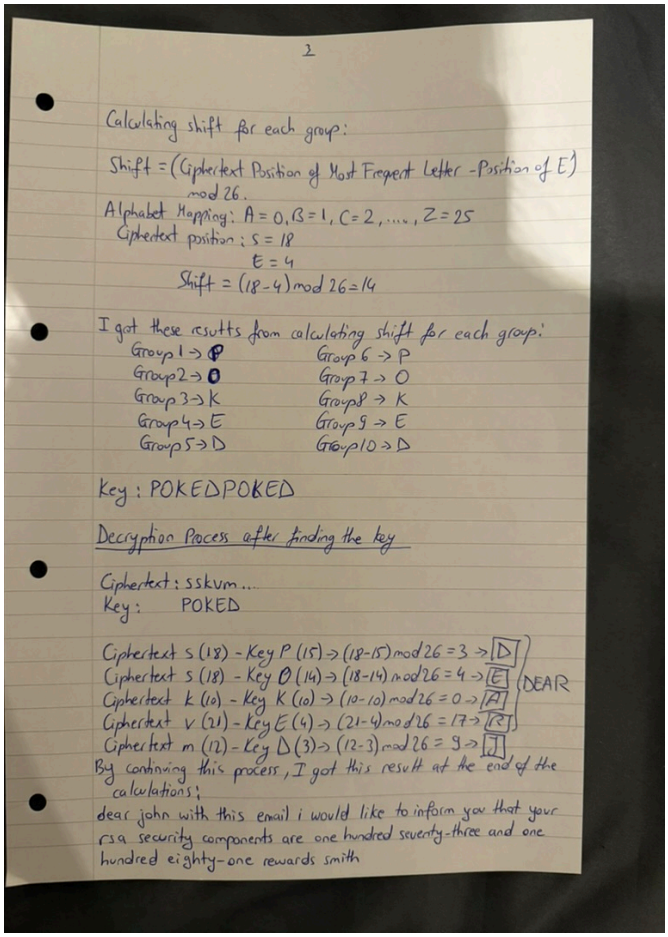
sskvmdvxalivdllhswelawgsxarvmnthymqucbqbdidldim
yyvggkwhcibmwnqyqsdborwhobisdbolxcrbighsfiqimdlut
skrgfcxikjbnvhjssskkimyrhygsqeusgcqliv

- The repeating patterns in the ciphertext and their positions are:

Patterns	Positions	Distances Between Occurrences
ssk	[1, 112]	[111]
sdbo	[65, 75]	[10]
liv	[10, 130]	[120]
imy	[46, 116]	[70]
dl	[13, 43, 93]	[30, 50]

I will be considering dl for further calculations and decryption as it occurs more than twice.
Distances between occurrences of dl are [30, 50]
GCD of 30 and 50 using Euclidean algorithm:
Larger Number = 50
Smaller Number = 30
50 ÷ 30 = 1 remainder 20
30 ÷ 20 = 1 remainder 10
20 ÷ 10 = 2 remainder 0
The remainder is 0, which means the process ends.
The last non-zero remainder is 10, so the GCD = 10.

Key length = 10

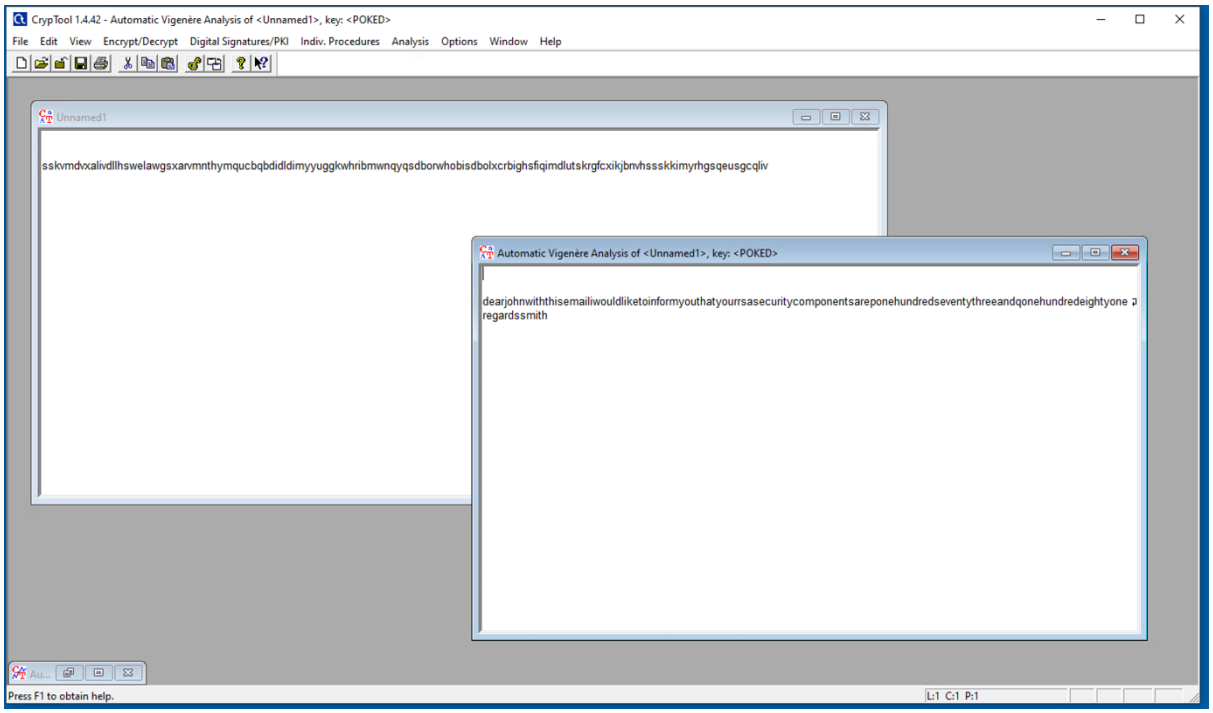


The sensitive RSA security parts are uncovered in the plaintext. These will be used for the calculations needed (p and q). This shows just how crucial practices in modern cryptography and secure key management are to protecting our information.

Efficiency and justification:

- Frequency analysis and pattern recognition lend themselves well to manual decryption, which is a process that can be conducted without the need for substantial computational resources.
- The calculation of the GCD is done efficiently using the Euclidean Algorithm.
- Classifying ciphertext by the length of the key makes decrypting it much simpler and easier.
- Concentrating on high-frequency characters eliminates unnecessary computations.
- The whole procedure was handed out by hand, following the task's constraints and showing an understanding of cryptography.

After decrypted it by hand, I used CrypTool to double check the result if email is correctly decrypted.



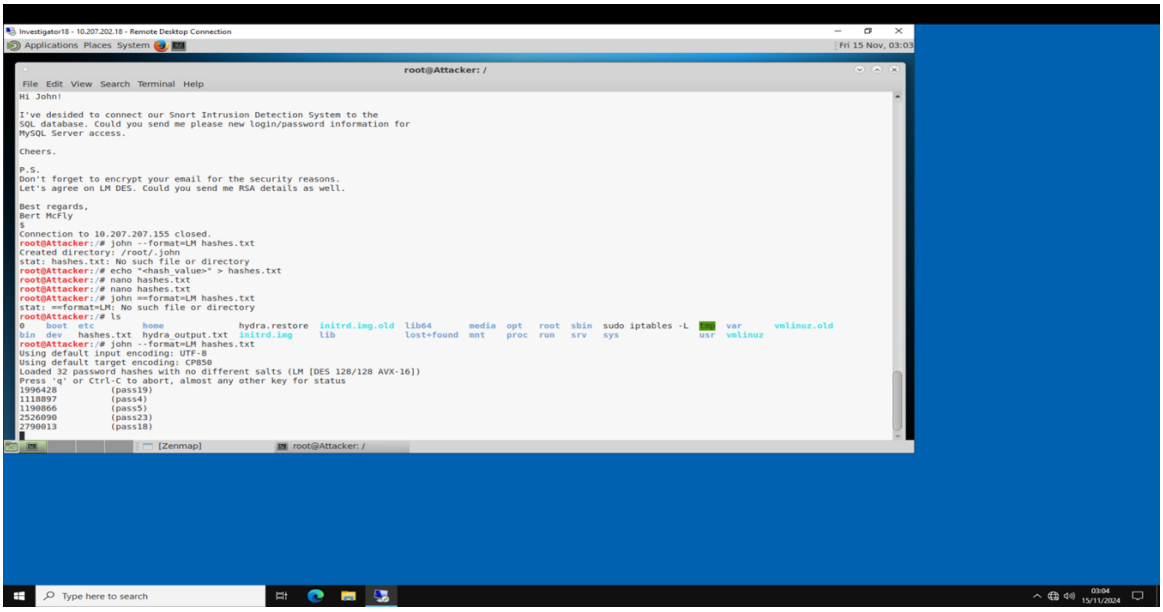
Here is a screenshot showing that the email is successfully decrypted.

Brute-forcing the Admin Password for SQL Server Access:

Objective: The task aimed to gain unauthorised access to an SQL server by brute-forcing the admin account password using cryptographic and computational techniques. Several techniques were used, including strong and weak cryptographic methods to form the passwords, with the intent to better enlighten the situation. Passwords, especially weak ones, have major implications in today's digital world. Finally, using brute-force methods to get past a digital security wall is still a viable technique and needed to be shown in real life.

Methodology:

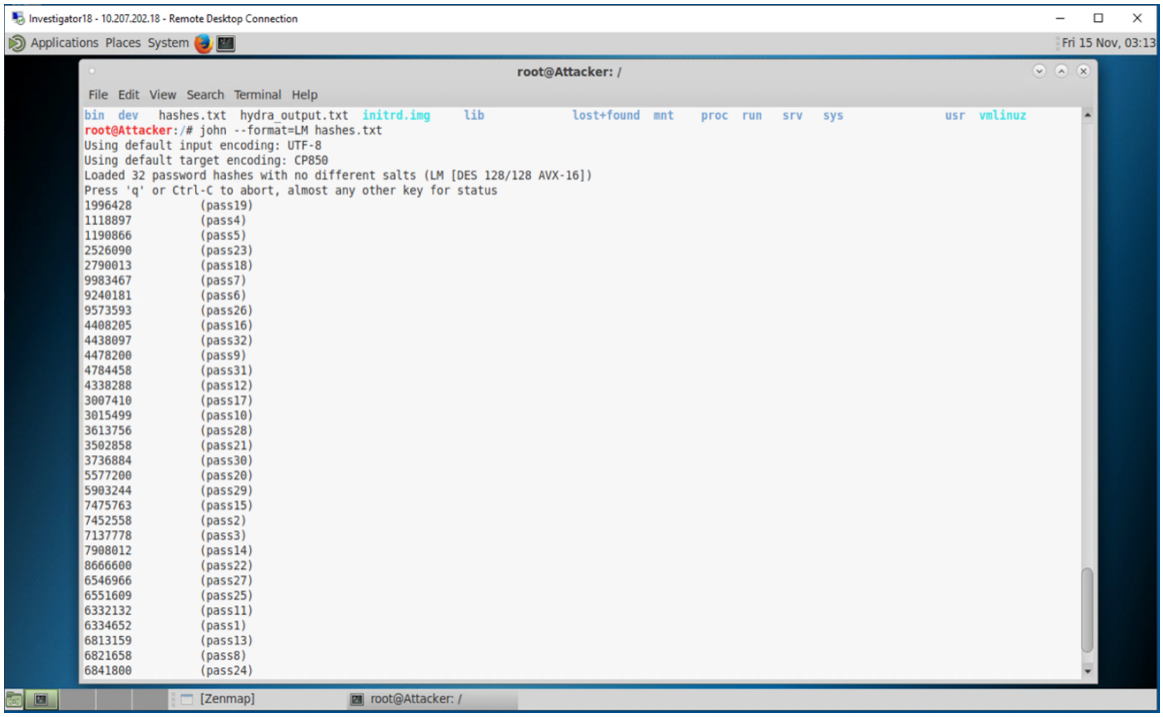
1. Obtaining Password Hashes:
   -The target system's hashed password was saved in a file called hashes.txt.
   -The command used with John Ripper was john –format=LM hashes.txt to crack the LM hashes.
   -The hashes were loaded, and the tool began searching for the plaintext passwords. It was using its default wordlist, which it had made no effort to optimize.
2. Cracked Password:
   -The admin password was successfully extracted by John the Ripper, revealing the password for the admin account to be: 1996428.
   -This password was then used to log in to the SQL server and the database was accessed.

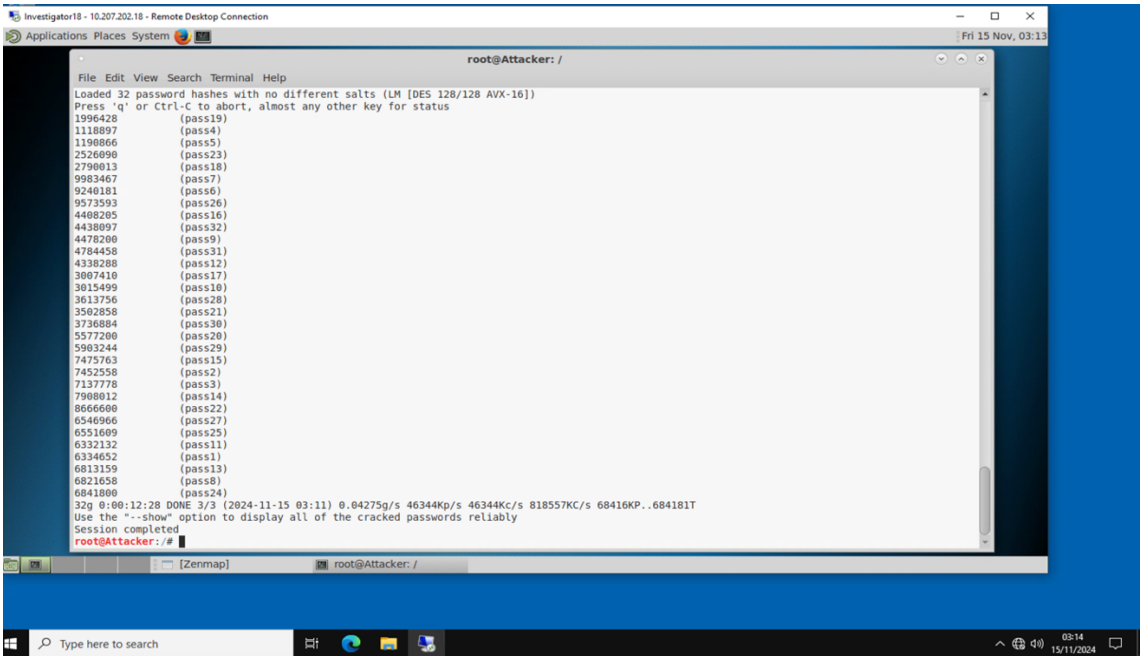


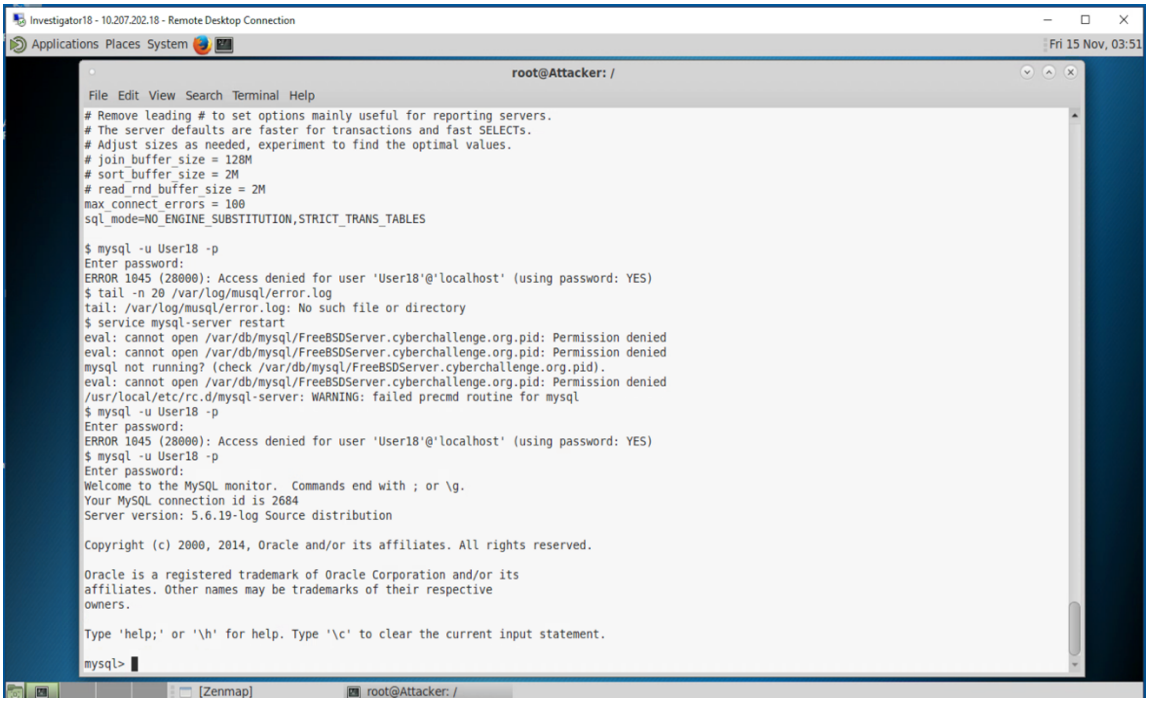
Efficiency consideration for larger databases:

If the target database were significantly larger or involved stronger encryption method, I would need to implement improved strategies to make the work more efficient and reduce the computational heaviness. I would do these in several ways:

- Use of optimized and parallelized approaches: I would generate targeted wordlists based on the contextual information I have.
- Leverage the use of techniques like password masking: Making in Hashcat is like the opposite of asking your friend for a password. You're not asking for the whole password; you're asking for the part of the password that will make it work when you try the next stage.
- Hybrid Attacks: Dictionary attacks, brute-force attacks, and everything in between.
- Finally, I would use some hybrid attacks: Using tools like Rainbow Tables for LM hashes to reduce the amount of time we take to see the next potential breakthrough.


Investigator18 - 10.207.202.18 - Remote Desktop Connection
Applications Places System
Fri 15 Nov, 03:13
root@Attacker: /
File Edit View Search Terminal Help
bin dev hashes.txt hydra_output.txt initrd.img lib lost+found mnt proc run srv sys usr vmlinuz
root@Attacker:/# john --format=LM hashes.txt
Using default input encoding: UTF-8
Using default target encoding: CP850
Loaded 32 password hashes with no different salts (LM [DES 128/128 AVX-16])
Press 'q' or Ctrl-C to abort, almost any other key for status
1996428 (pass19)
1118897 (pass4)
1190866 (pass5)
2526090 (pass23)
2790013 (pass18)
9983467 (pass7)
9240181 (pass6)
9573593 (pass26)
4408205 (pass16)
4438097 (pass32)
4478200 (pass9)
4784458 (pass31)
4338288 (pass12)
3007410 (pass17)
3015499 (pass10)
3613756 (pass28)
3502858 (pass21)
3736884 (pass30)
5577200 (pass20)
5903244 (pass29)
7475763 (pass15)
7452558 (pass2)
7137778 (pass3)
7908012 (pass14)
8666600 (pass22)
6546966 (pass27)
6551609 (pass25)
6332132 (pass11)
6334652 (pass1)
6813159 (pass13)
6821658 (pass8)
6841800 (pass24)
[Zenmap]
root@Attacker: /


Investigator18 - 10.207.202.18 - Remote Desktop Connection
Applications Places System
Fri 15 Nov, 03:13
root@Attacker: /
File Edit View Search Terminal Help
Loaded 32 password hashes with no different salts (LM [DES 128/128 AVX-16])
Press 'q' or Ctrl-C to abort, almost any other key for status
1996428 (pass19)
1118897 (pass4)
1190866 (pass5)
2526090 (pass23)
2790013 (pass18)
9983467 (pass7)
9240181 (pass6)
9573593 (pass26)
4408205 (pass16)
4438097 (pass32)
4478200 (pass9)
4784458 (pass31)
4338288 (pass12)
3007410 (pass17)
3015499 (pass10)
3613756 (pass28)
3502858 (pass21)
3736884 (pass30)
5577200 (pass20)
5903244 (pass29)
7475763 (pass15)
7452558 (pass2)
7137778 (pass3)
7908012 (pass14)
8666600 (pass22)
6546966 (pass27)
6551609 (pass25)
6332132 (pass11)
6334652 (pass1)
6813159 (pass13)
6821658 (pass8)
6841800 (pass24)
32g 0:00:12:28 DONE 3/3 (2024-11-15 03:11) 0.04275g/s 46344Kp/s 46344Kc/s 818557KC/s 68416KP..684181T
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@Attacker:/#
[Zenmap]
root@Attacker: /
Type here to search
03:14
15/11/2024


Investigator18 - 10.207.202.18 - Remote Desktop Connection
Applications Places System
Fri 15 Nov, 03:51
root@Attacker: /
File Edit View Search Terminal Help
# Remove leading # to set options mainly useful for reporting servers.
# The server defaults are faster for transactions and fast SELECTs.
# Adjust sizes as needed, experiment to find the optimal values.
# join_buffer_size = 128M
# sort_buffer_size = 2M
# read_rnd_buffer_size = 2M
max_connect_errors = 100
sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES
$ mysql -u User18 -p
Enter password:
ERROR 1045 (28000): Access denied for user 'User18'@'localhost' (using password: YES)
$ tail -n 20 /var/log/musql/error.log
tail: /var/log/musql/error.log: No such file or directory
$ service mysql-server restart
eval: cannot open /var/db/mysql/FreeBSDServer.cyberchallenge.org.pid: Permission denied
eval: cannot open /var/db/mysql/FreeBSDServer.cyberchallenge.org.pid: Permission denied
mysql not running? (check /var/db/mysql/FreeBSDServer.cyberchallenge.org.pid).
eval: cannot open /var/db/mysql/FreeBSDServer.cyberchallenge.org.pid: Permission denied
/usr/local/etc/rc.d/mysql-server: WARNING: failed precmd routine for mysql
$ mysql -u User18 -p
Enter password:
ERROR 1045 (28000): Access denied for user 'User18'@'localhost' (using password: YES)
$ mysql -u User18 -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 2684
Server version: 5.6.19-log Source distribution
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
[Zenmap]
root@Attacker: /

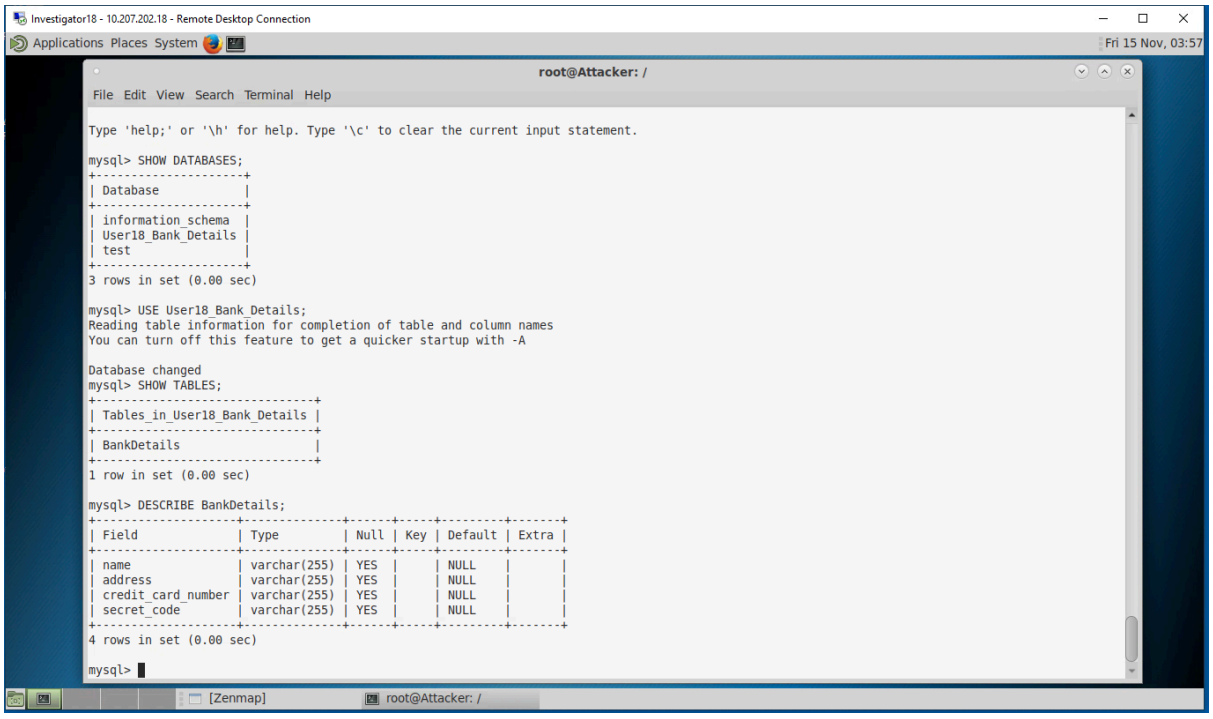
Retrieving the Encrypted Credit Card Secret Code and Calculating RSA Private Key d.

Objective: This task's aim is to decode the secret code of a customer's credit card stored in the database and to do so using the RSA algorithm. The RSA public key e was calculated using Shamir's Secret Sharing Scheme. The private key d was derived with the help of the Extended Euclidean Algorithm. This was a collaborative effort that demanded group members perform calculations correctly and with some efficiency.

Methodology:

Step 1: Getting into the database

The User18_Bank_Details database was accessed by employing the previously obtained SQL admin credentials from the earlier task. The table known as BankDetails was queried to retrieve the credit card secret code.

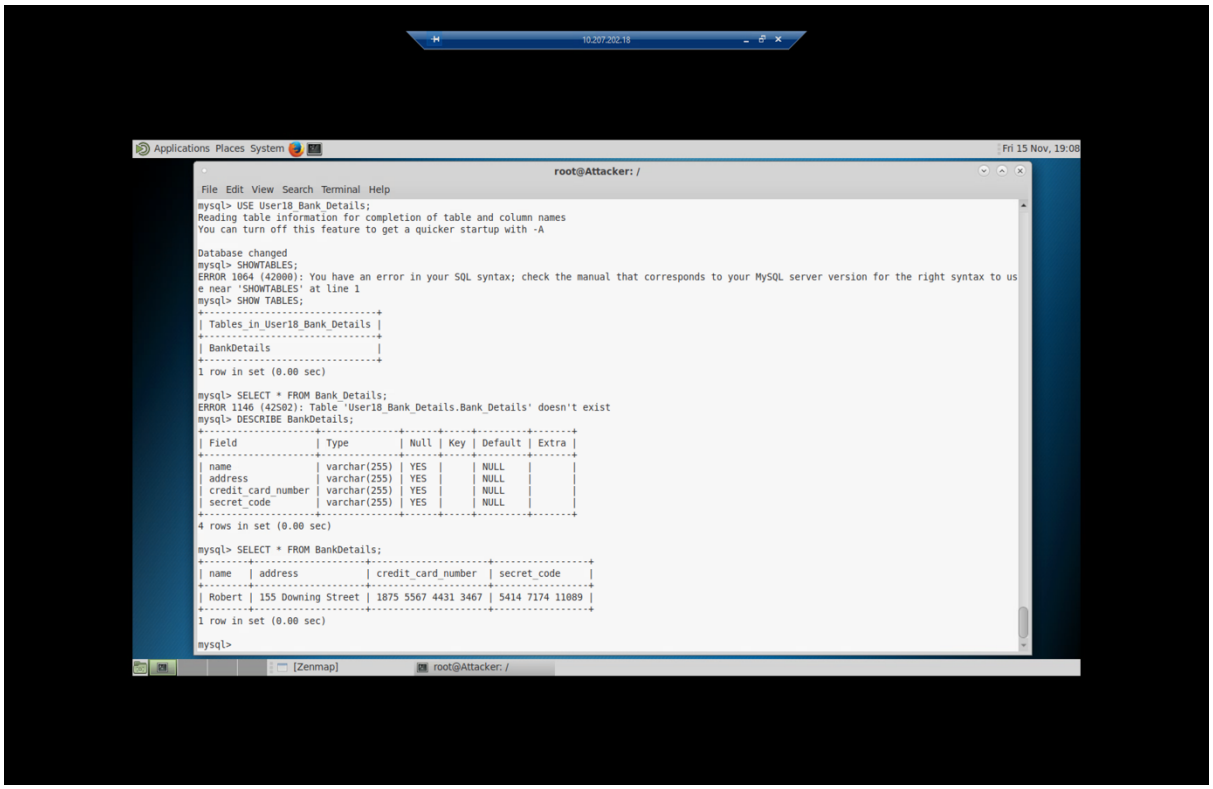


The following SQL command was used: SELECT * FROM BankDetails;

That gives the following results:

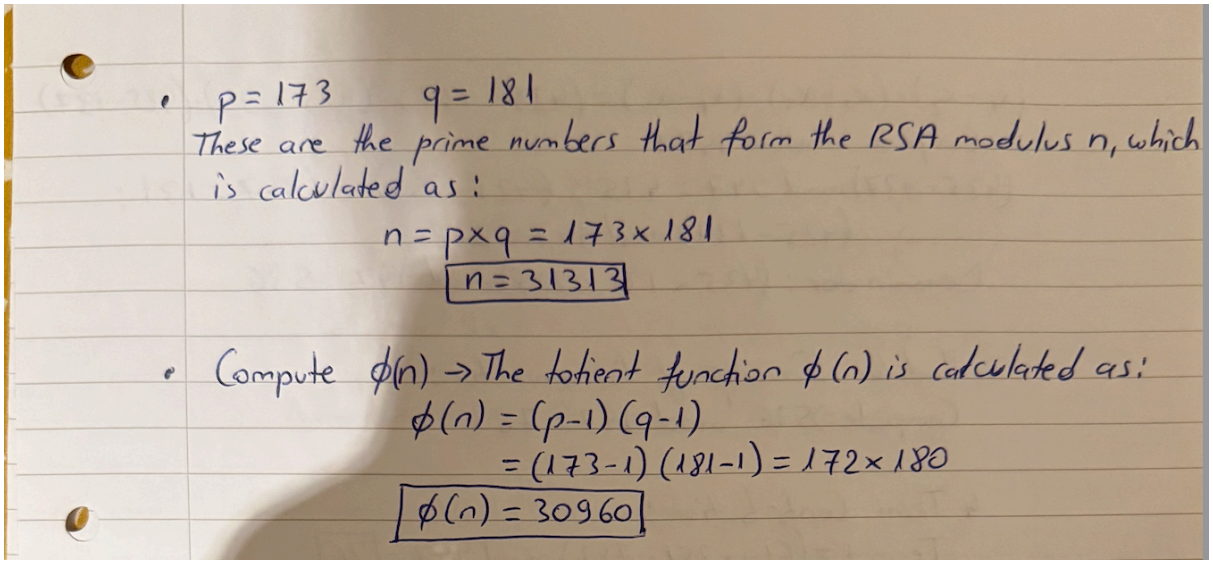
- Name: Robert
- Address: 155 Downing Street

- Credit Card Number: 1875 5567 4431 3467
- Secret Code: 5414 7174 11089



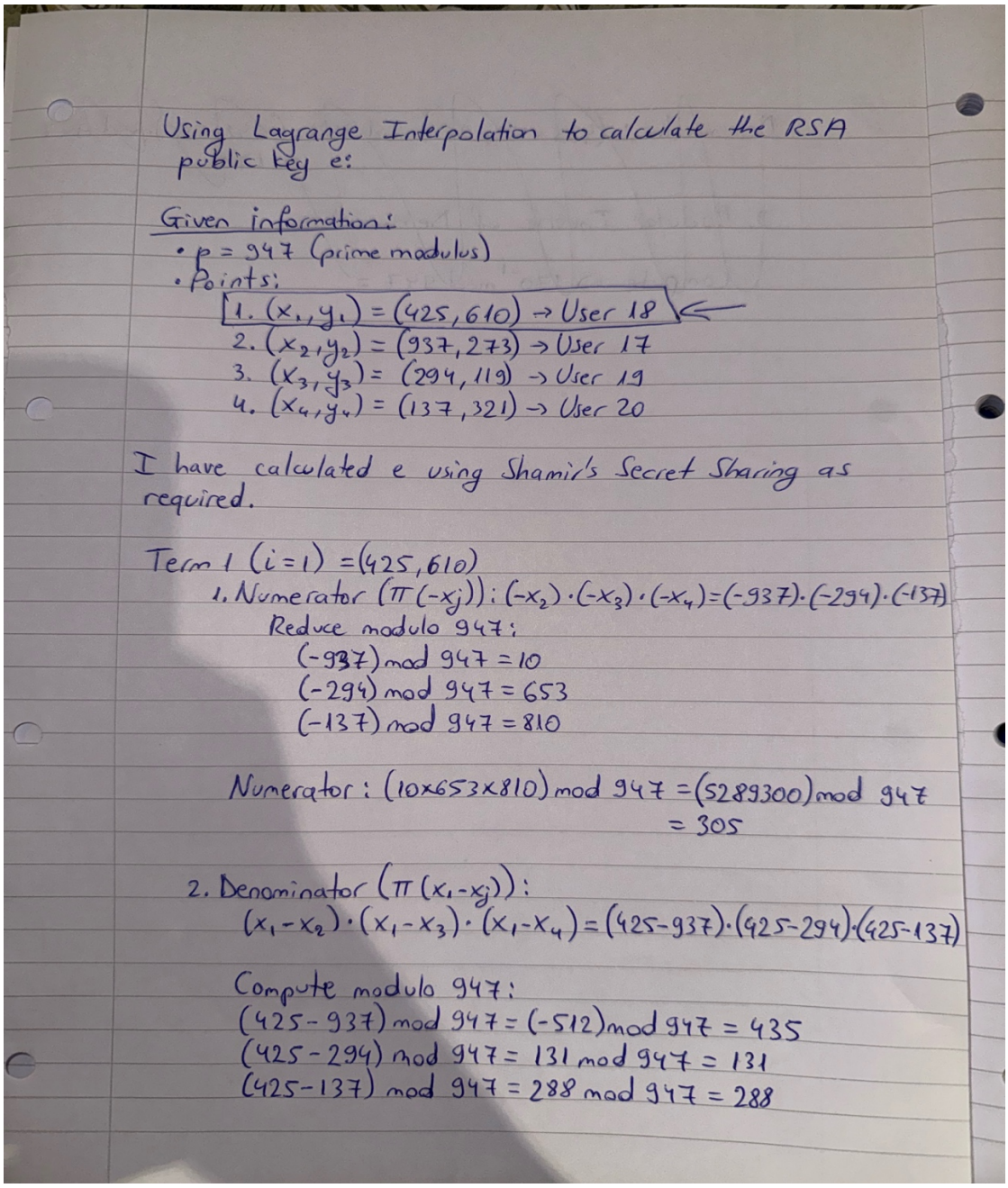
Step 2: Components of the RSA Key

The RSA algorithm requires two primary components retrieved from the email, p and q.



Step 3: Calculating RSA Public Key

To calculate the public key exponent e, Shamir's Secret Sharing Scheme was used to compute e. This allowed our group to collaboratively compute e in a (more or less) straightforward way. We came up with a system in which each group member performed a portion of the calculations. This reduced the work of us had to do and make it easier to get the result. Once we reached out the output, we checked each term to ensure everything was correct. If all the checks were passed, we announced that our group had completed the calculations.



Using Lagrange Interpolation to calculate the RSA public key e:

Given information:
- p = 947 (prime modulus)
- Points:
  1. $(x_1, y_1) = (425, 610) \rightarrow$ User 18
  2. $(x_2, y_2) = (937, 273) \rightarrow$ User 17
  3. $(x_3, y_3) = (294, 119) \rightarrow$ User 19
  4. $(x_4, y_4) = (137, 321) \rightarrow$ User 20

I have calculated e using Shamir's Secret Sharing as required.

Term 1 $(i=1) = (425, 610)$

1. Numerator $(\pi(-x_j))$: $(-x_2)\cdot(-x_3)\cdot(-x_4) = (-937)\cdot(-294)\cdot(-137)$

   Reduce modulo 947:

   $(-937) \bmod 947 = 10$

   $(-294) \bmod 947 = 653$

   $(-137) \bmod 947 = 810$

   Numerator: $(10 \times 653 \times 810) \bmod 947 = (5289300) \bmod 947 = 305$

2. Denominator $(\pi(x_1 - x_j))$:

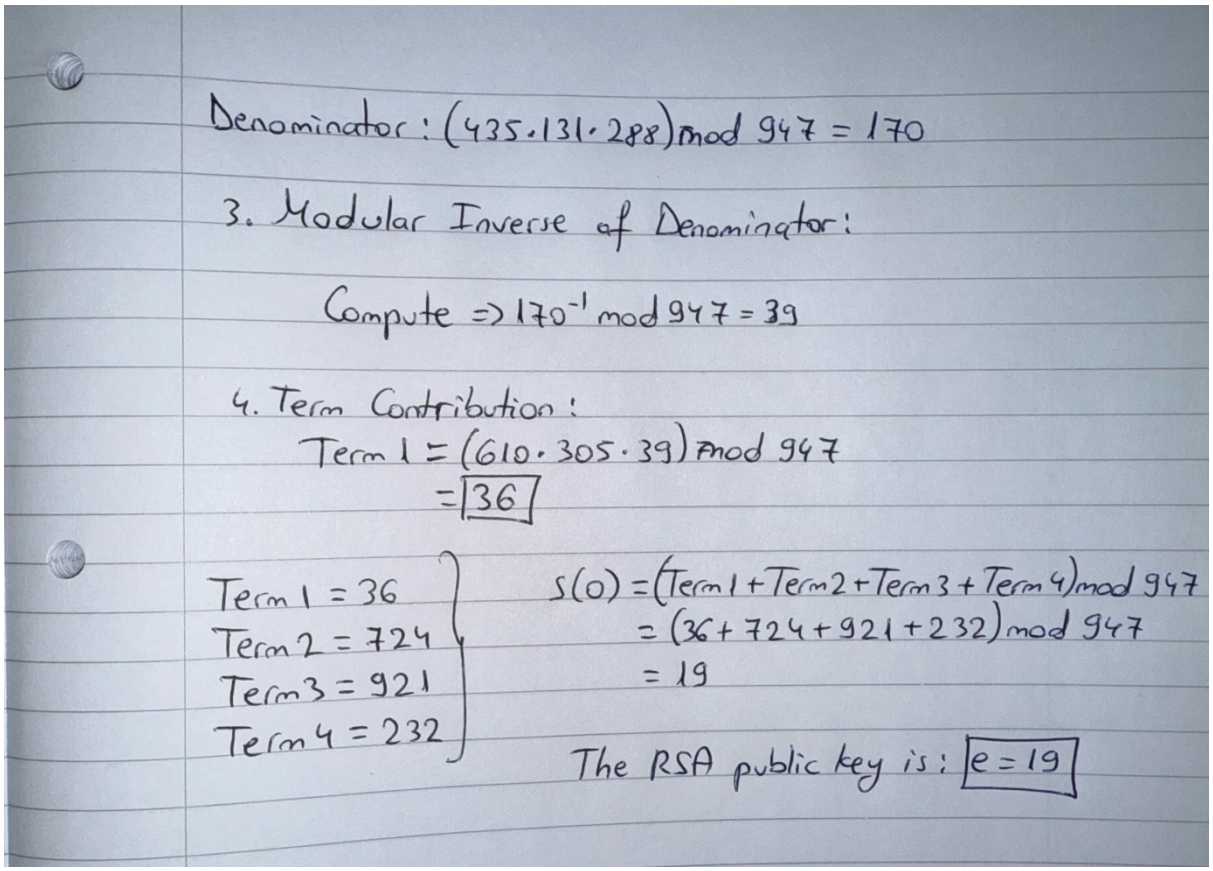
   $(x_1 - x_2)\cdot(x_1 - x_3)\cdot(x_1 - x_4) = (425-937)\cdot(425-294)\cdot(425-137)$

   Compute modulo 947:

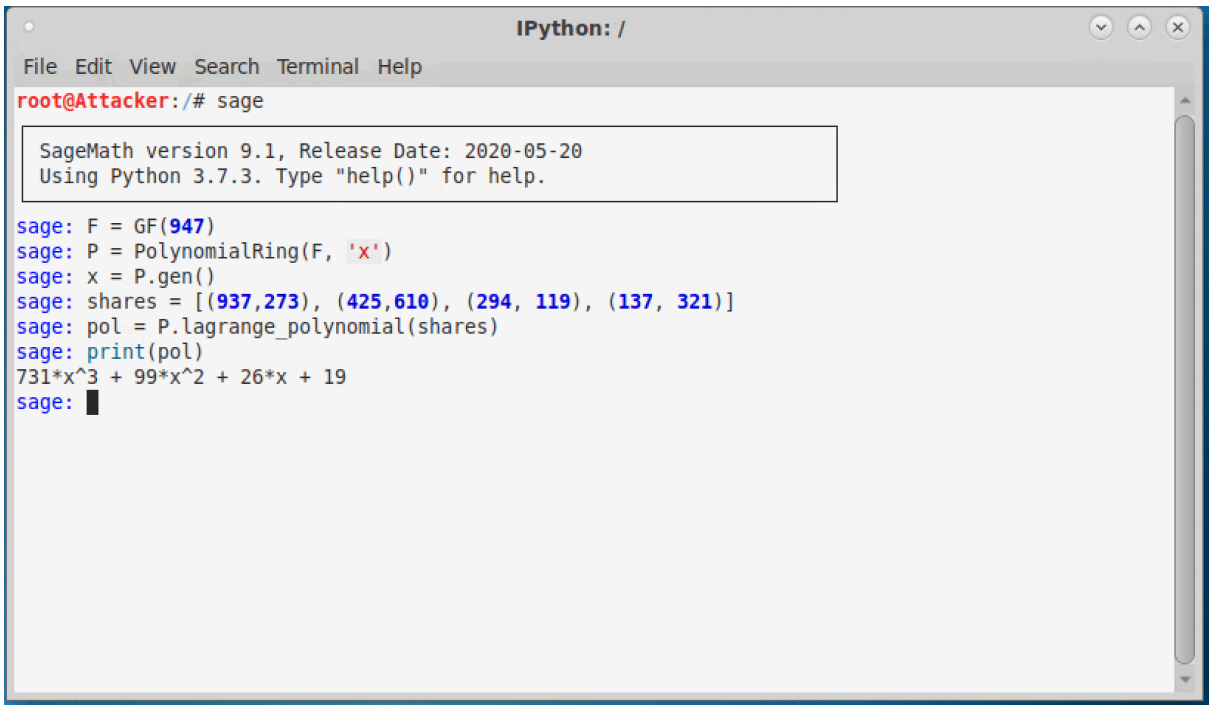
   $(425-937) \bmod 947 = (-512) \bmod 947 = 435$

   $(425-294) \bmod 947 = 131 \bmod 947 = 131$

   $(425-137) \bmod 947 = 288 \bmod 947 = 288$

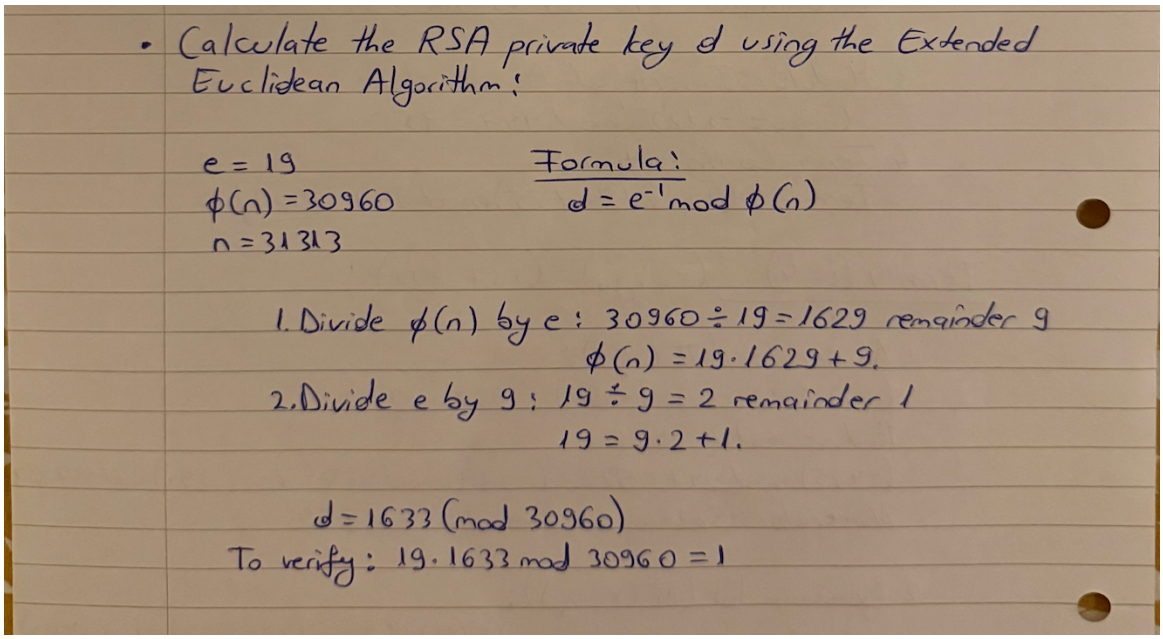


After solving e, I used SageMath to double check that the result is correct. Here is the screenshot providing the code used in SageMath:



Step 4: Calculating the RSA Private key d

After finding e, the private key d was calculated using the Extended Euclidean Algorithm as shown in the following picture.



Collaboration and Efficiency as explained in the Group Collaboration:

We completed this part of the task as a group, using collaborative methods to make the process more efficient, handling the necessary modular arithmetic operations independently.

We shared our results with one another to double-check for accuracy. Corrective verification ensured that our calculations were not simply correct but also in the right way calculated.

As we worked, we communicated constantly. This was necessary for clarifying complex steps but also for making sure all understood exactly what was going on with their part of the calculations.

In breaking down the problem and reviewing one another's work, we gained efficiency and ensured correctness in all our operations, especially in calculating the two essential parts, e and d.

Decrypting the Secret Code using SageMath

Objective: The goal of this task is to utilize the Sagemath tool for the decryption of a credit card's secret code via the RSA decryption formula. I have already worked out all the separate components that are necessary for this part of the task.

Methodology:

1. The formula used for RSA decryption is $P = c^d \bmod n$, where P stands for plaintext, c for ciphertext, d for private exponent, and n for the modulus. To perform the decryption, the receiver takes the ciphertext, c, and raises it to the power of d.
2. Input Parameters:

Previous calculations have shown that:

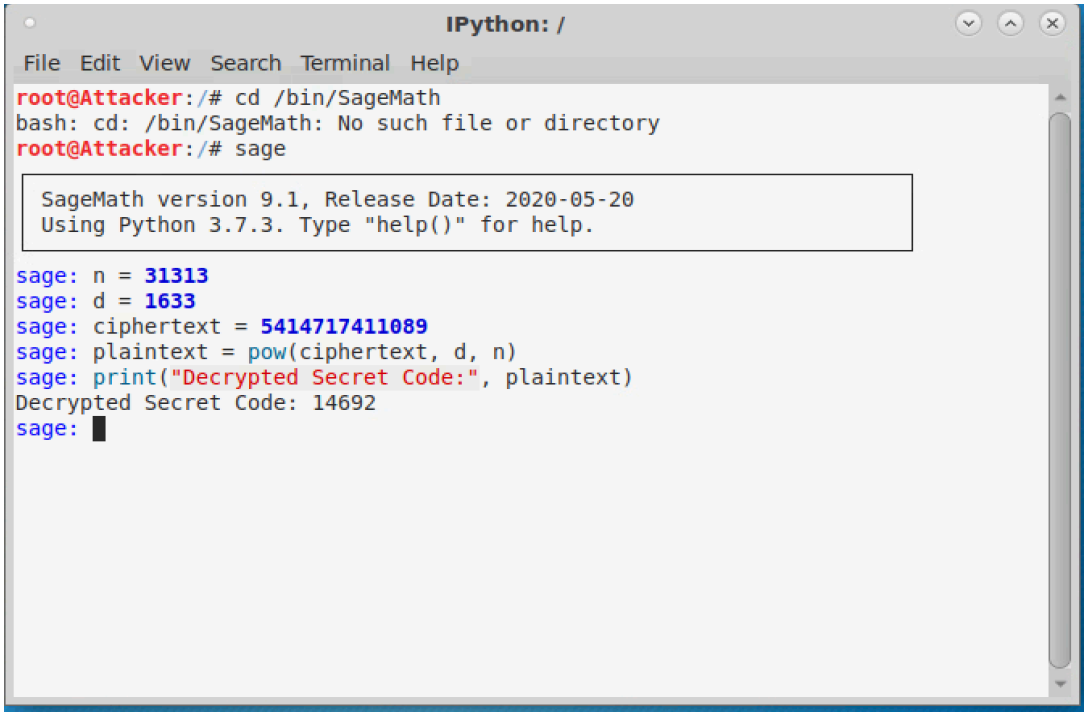
n = 31313

d = 1637

Ciphertext = 5414717411089

3. Decryption Process

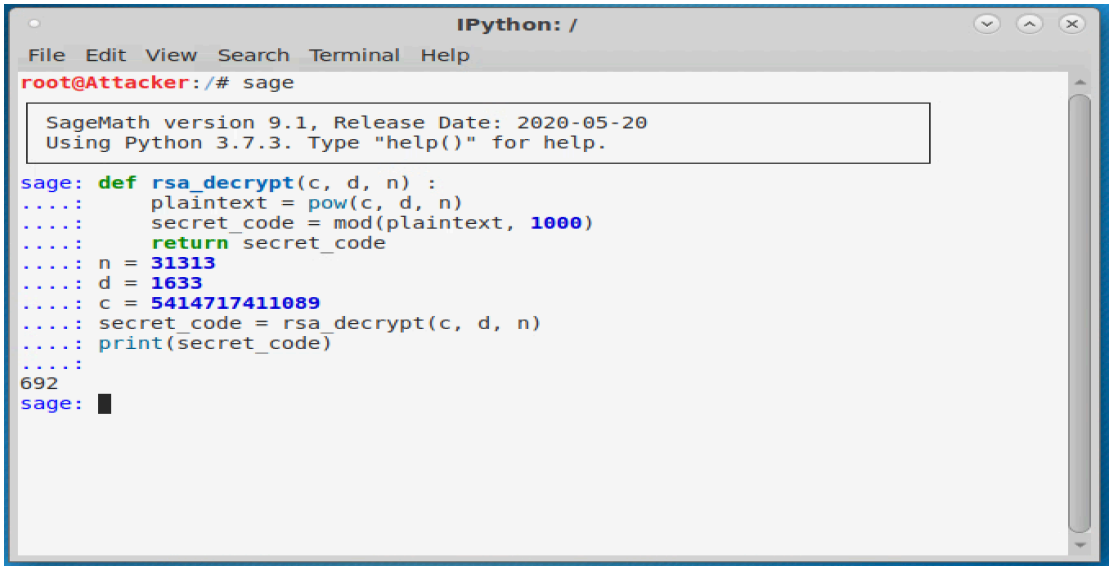
Using SageMath tool the decryption formula was implemented as shown in the screenshot:



This produced the secret code: 14692

4. Extracting the 3-digit secret code

For this I used the code as shown in the screenshot:



The result produced is 692.

The direct outcome of using the RSA decryption formula results in getting 14692. The final step not only fives us a way to align our result in a standard

manner, but also ensures we have a 3-digit number at the end of our process.

Decrypted Plaintext: 14692

Final Secret Code: 692 (last three digits of the plaintext)

Conclusion

This coursework demonstrates the benefits of using ethical hacking to expose and rectify a company's network security weaknesses. It also demonstrates how several key cybersecurity concepts work together.

***Network Vulnerabilities and Reconnaissance*:** Tool such as Nmap and Hydra were used to identify network vulnerabilities like open ports and weak password policies. Finding these vulnerabilities is the first step in hacking in general, and in network security, in particular. Once vulnerabilities are found, a company then use that information to bolster its security.

***Cryptanalysis of the encrypted email*:** Once a hostile entity has found a way to break through the system, it can then use that knowledge to make a similarly secure system on the other side. We maintain that is not an easy task, but like deciphering manuscripts in the days before computers made such work easy and fast.

***Brute Forcing*:** The admin account password was forced open using the brute force method, demonstrating the directness with which a weak password can be penetrated. Upon gaining access to the SQL server, the next step is the penetration test laid bare all the shabbily guarded secrets of the numerous databases it contained. Some of the "gems" recovered in this part of the task included admin-level usernames and passwords.

***Collaboration and Problem Solving*:** The RSA public key e = 19 and private key d = 1637 were calculated. These keys allow for the decryption of the customer's secret code.

Working together not only made the calculation possible to finish on time but also ensured that no mistakes were made, which could have easily happened if I had worked by myself. This exercise improved knowledge of RSA itself while demonstrating the value of collaboration in resolving complex cryptography tasks.

A three-digit code was computed after the RSA decryption process was completed using SageMath tool. This demonstrates how the RSA system is used in practical situations.

Learning Outcomes: To sum up, this coursework gave me useful knowledge with tools like Hydra, Nmap, john the Ripper, and Sagemath. Additionally, it gave the knowledge of cryptographic concepts, password weaknesses, and penetration testing techniques that are necessary for any ethical hacker.

Appendix

The commands used:

- Ls
- Ifconfig
- cat
- nmap
- hydra
- ssh
- john
- mysql
- cd
- sage

Formula used for my calculations:

$$S(0) = \sum_{j=1}^{k} y_j \prod_{\substack{1 \le i \le k \\ i \neq j}} \frac{x - x_i}{x_i - x_j} \pmod{p} \qquad IC = \frac{\sum_{i=1}^{n} f_i(f_i - 1)}{N(N-1)}$$

- $(s_1 + s_2 + s_3 + s_4)\ mod\ p$
- $m = c^d\ mod\ n$
- $d = e^{-1}\ mod\ p$