Old Dominion University

CYSE 301 Cybersecurity Techniques and Operations

Assignment #5 – Password Cracking

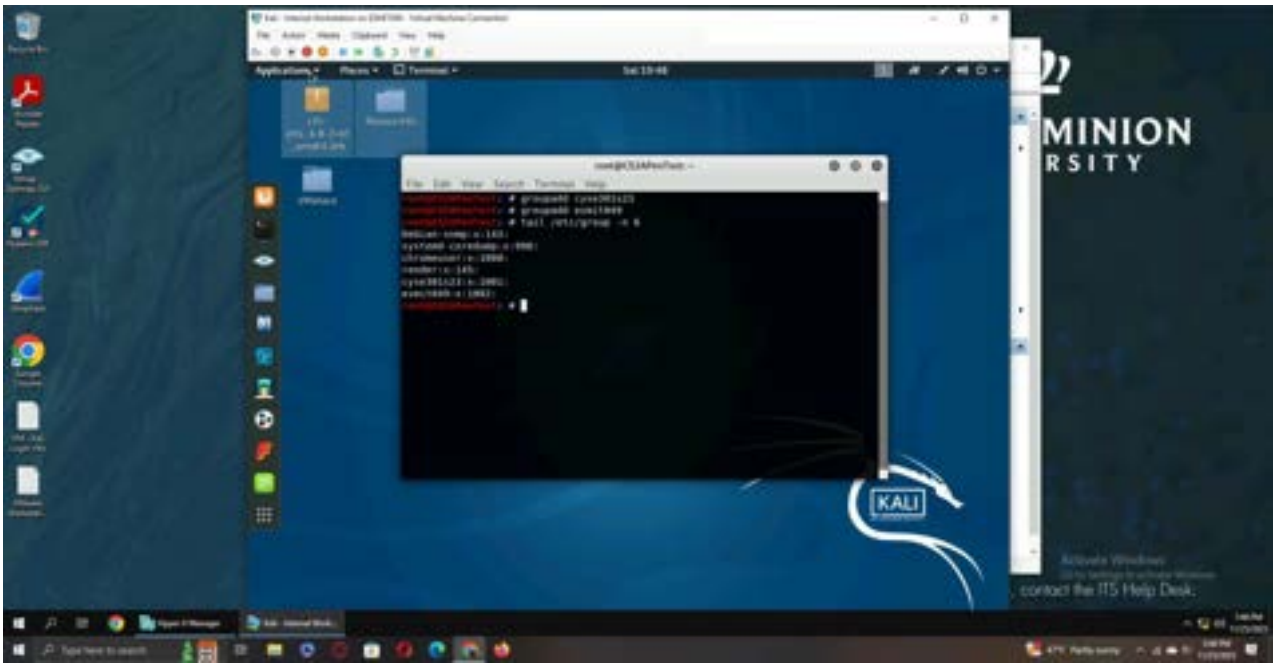
Ned Smith

01200384

Part One: Password Cracking

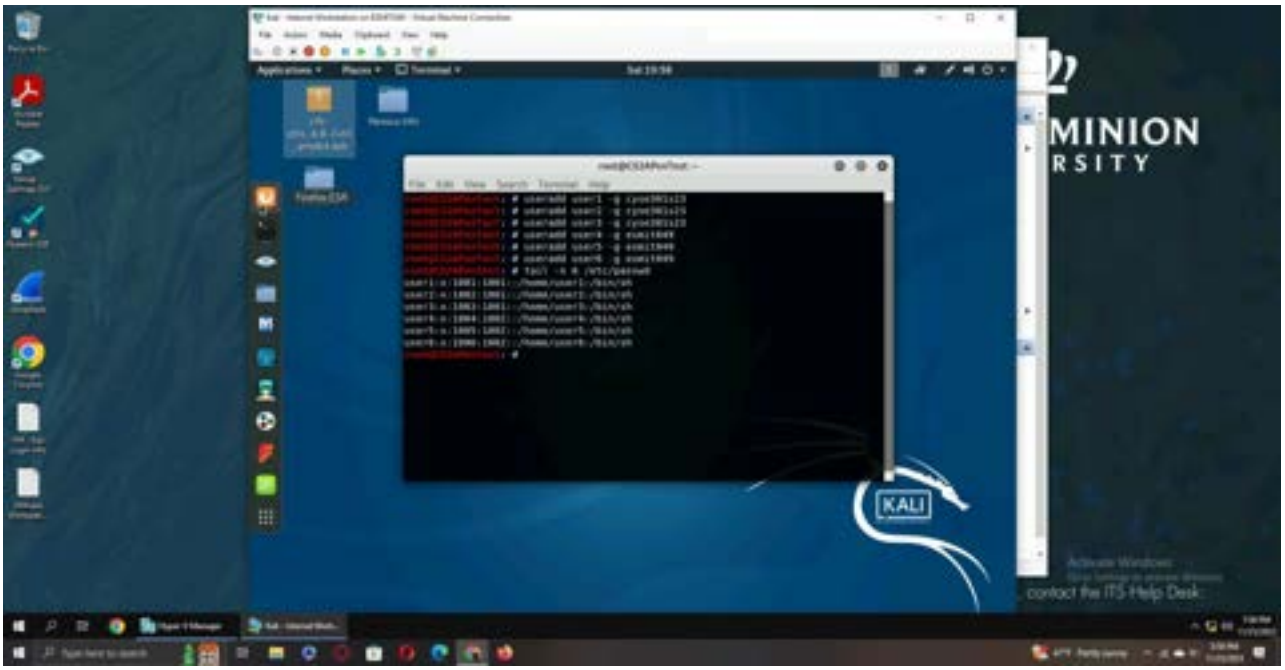
Task A: Linux Password Cracking

Group Creation



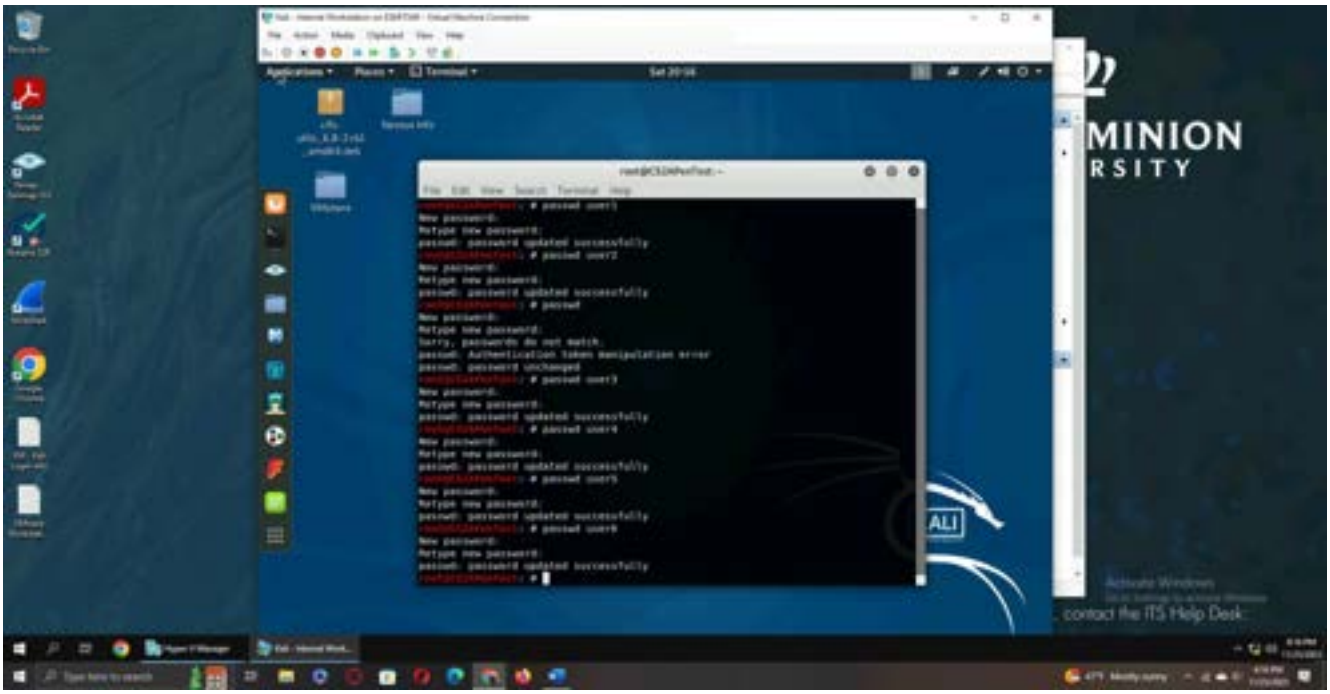
Explanation: I created two groups using the “groupadd” command, one named CYSE301s23 and the other named after my MIDAS ID.

User Creation



Explanation: I added six users, three to each of the groups, using the “useradd” command.

Password Creation



Explanation: Using the “passwd” command, I changed the passwords of each user to the following passwords:

User1: 321321

User2: abcd1234

User3: college

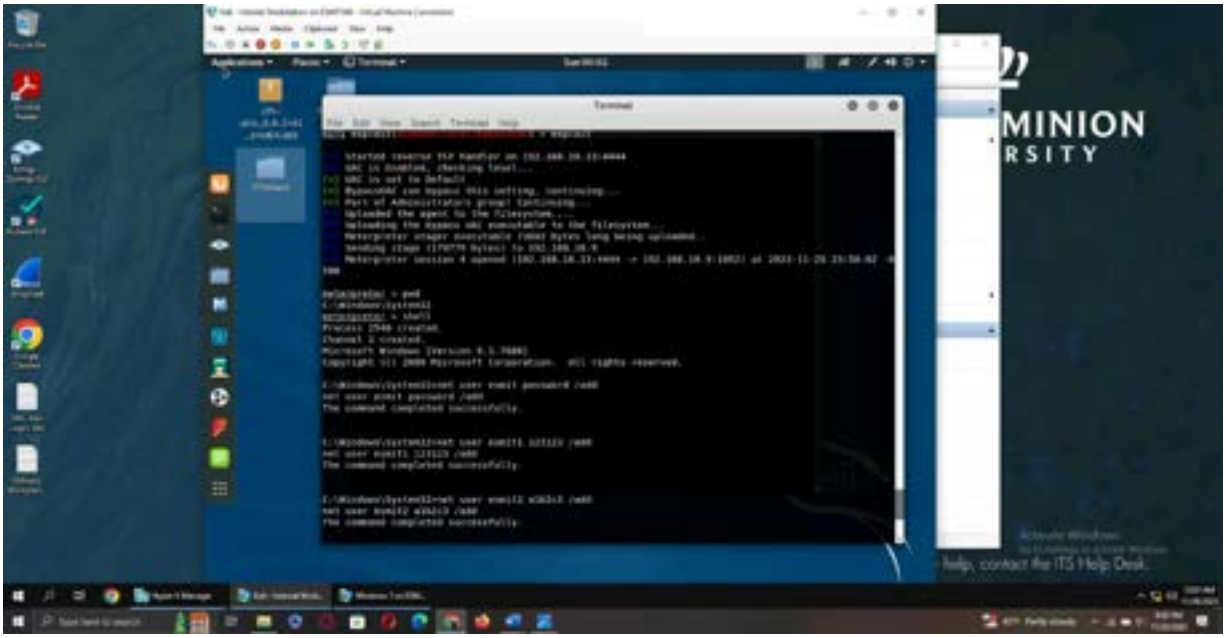
User4: a1b2c3

User5: SuperRad100%

User6: In2an3P4ssw0rd(*)

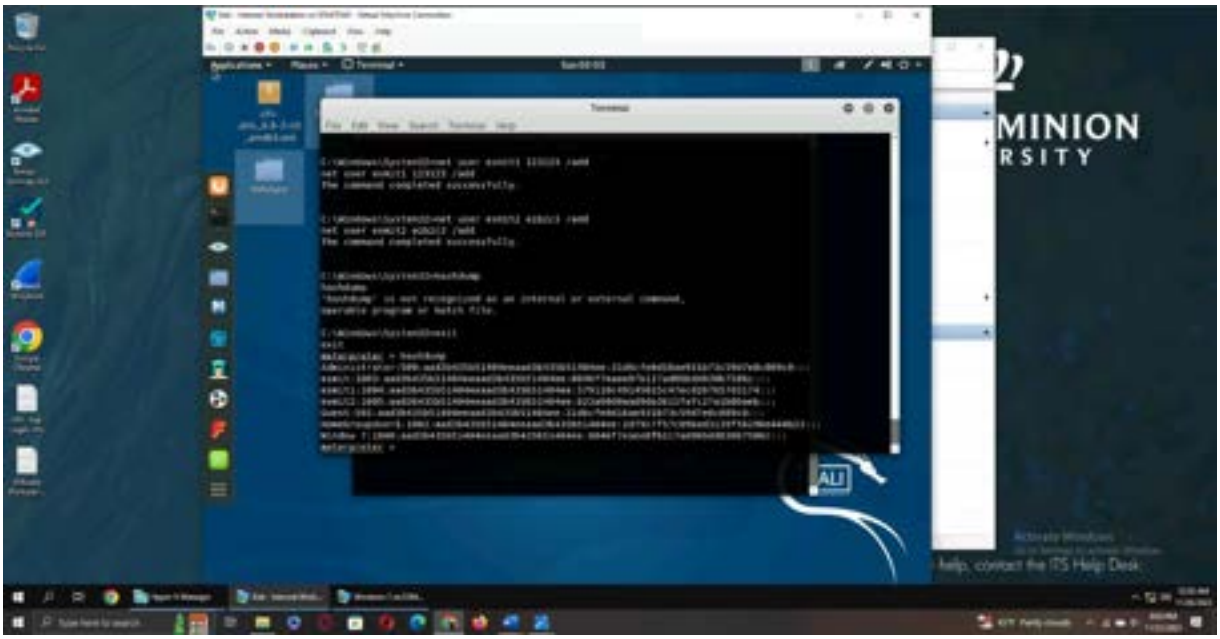
Task B: Windows Password Cracking

User Creation



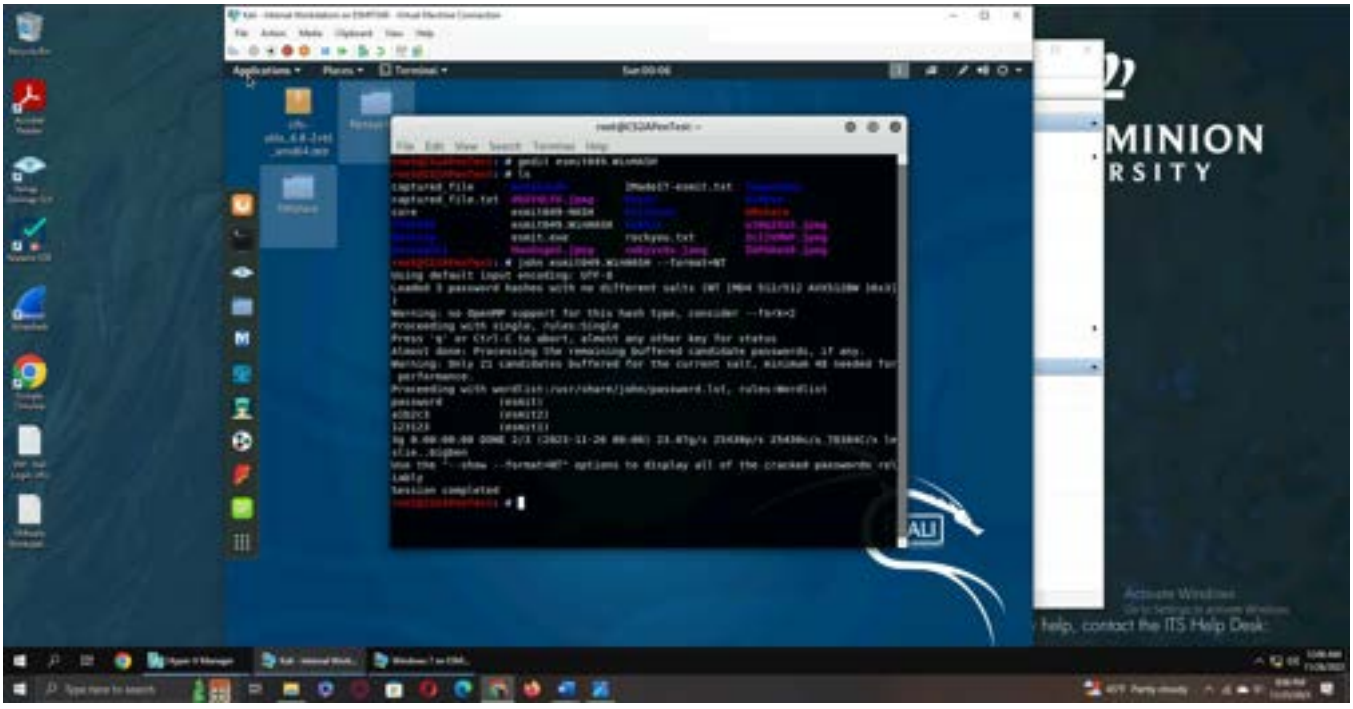
Explanation: After gaining administrator privileges and establishing a reverse shell connection, I used the net user command to add three users, as well as their passwords, to the Windows 7 VM.

Question 1



Explanation: I used the hashdump command in the meterpreter shell to display the password hashes

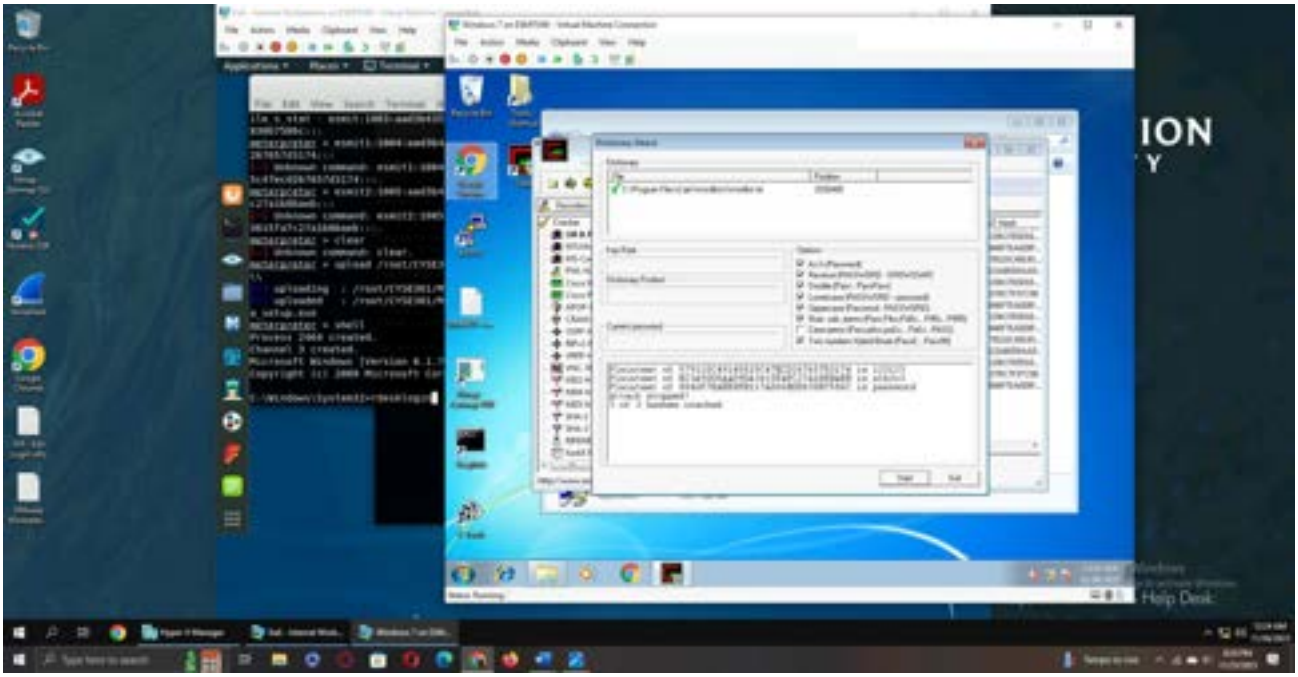
Question 2



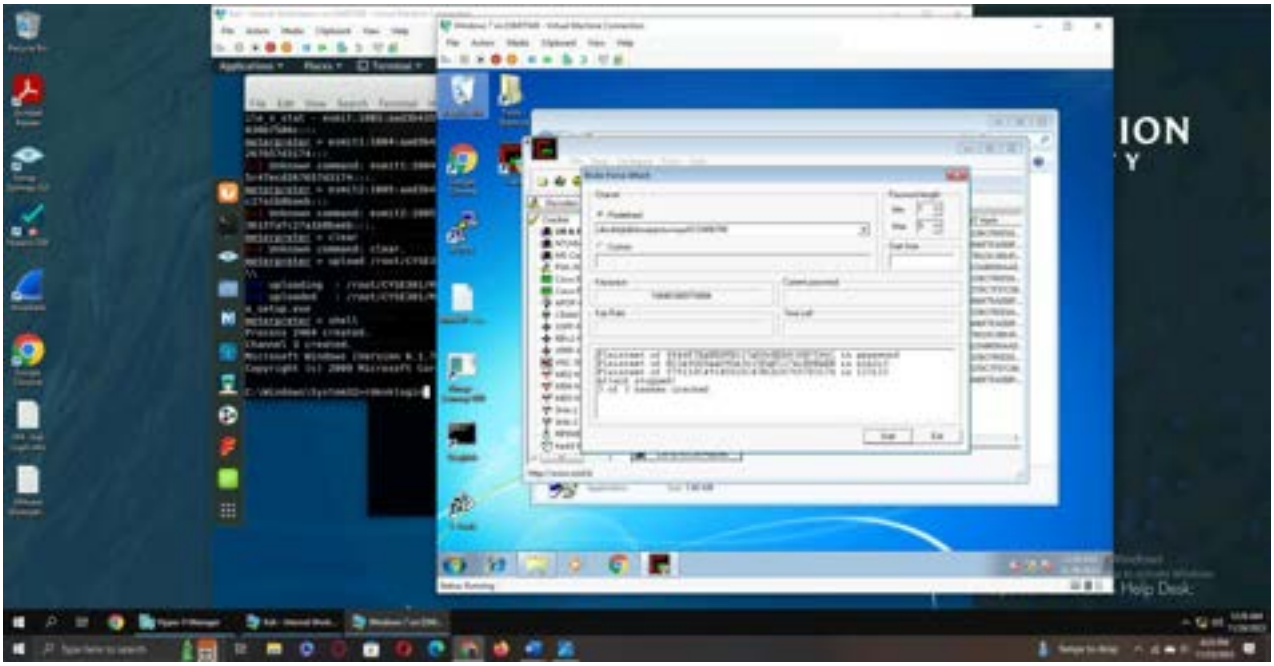
Explanation: Using the gedit command, I saved the hashes into a hash file. I then used John the Ripper to crack the three password hashes, with it finishing once it had cracked all three.

Question 3:

Dictionary Attack

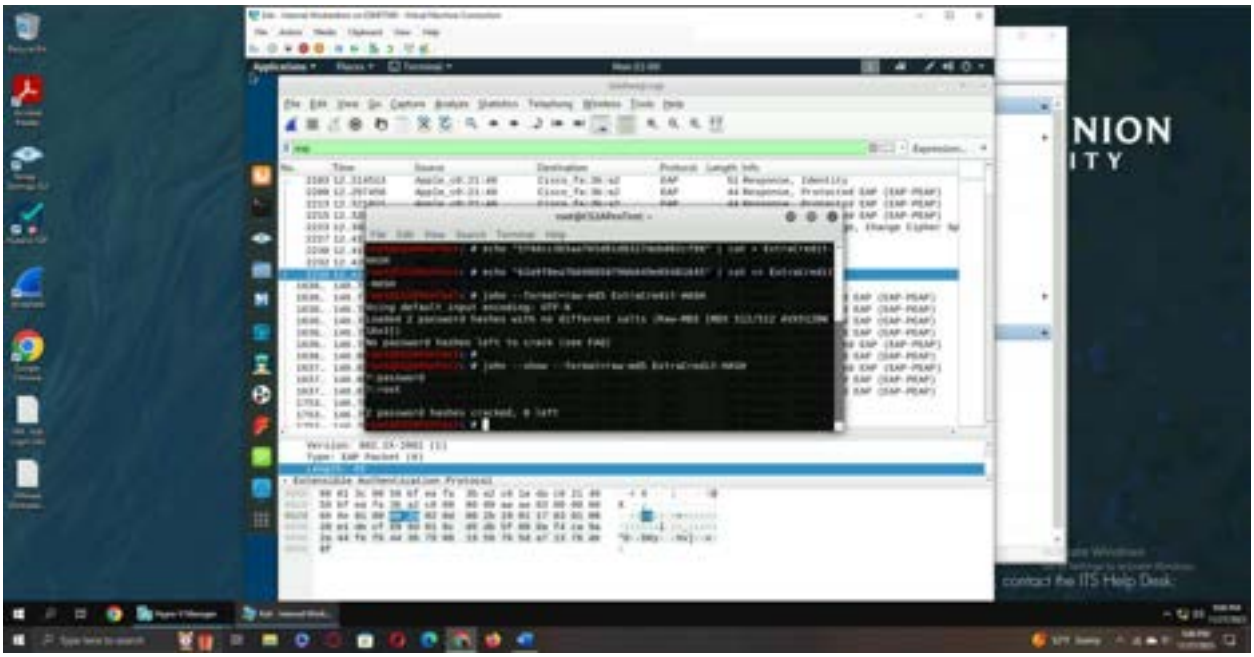


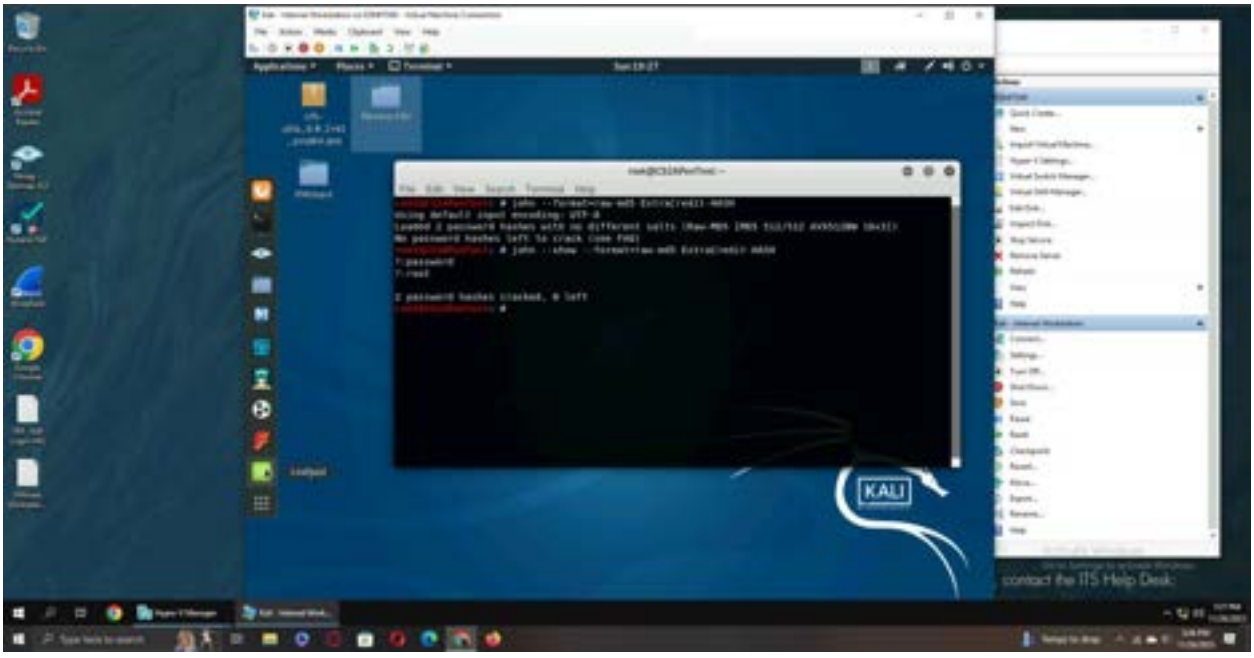
Brute Force Attack



Explanation: Using the previously mentioned reverse shell connection and administrator privileges, I uploaded the Cain and Abel password-cracking tool to the Windows 7 VM and installed it through the remote desktop window. I then used the Cain and Abel tool to run a dictionary attack as well as a brute force attack, with both resulting in all three hashes being cracked.

Task C: Extra Credit



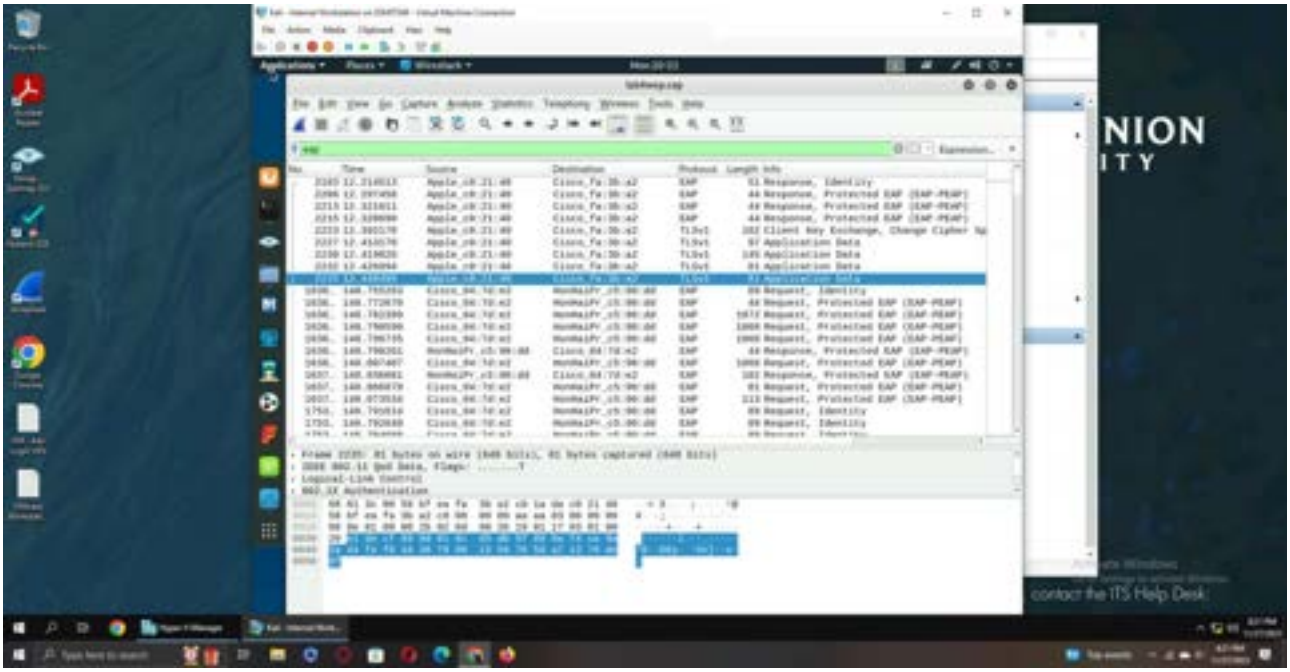


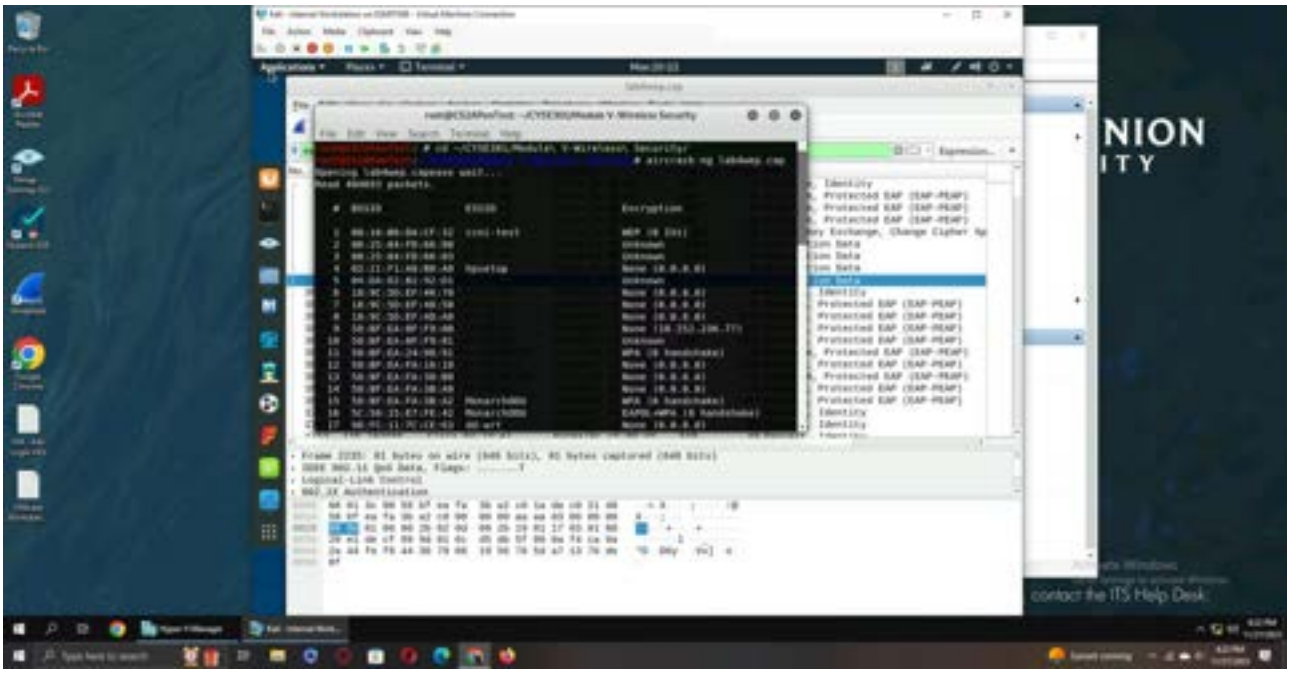
Explanation: I didn't save the initial steps leading up to cracking the hash originally, so the first screenshot is a recreation of how I saved the hashes in the hash file. Using a combination of the echo and cat commands, I wrote both hashes to the hash file called "ExtraCredit-HASH". I then used the format "raw-md5" with John the Ripper to crack the two passwords. The second screenshot is the original image showing that the passwords were cracked and displaying the two passwords, those being password and root. I don't have a screenshot of the original password-cracking session with John the Ripper, and it won't perform an additional one because the cracked hashes are saved by John the Ripper.

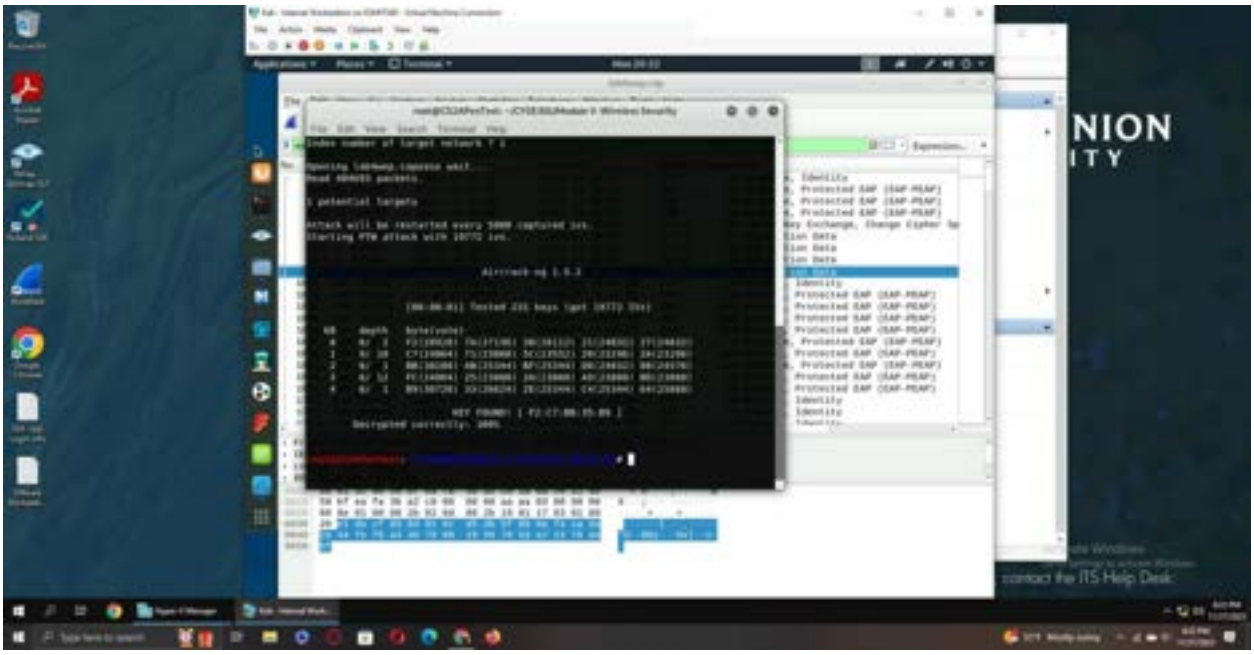
Part Two: Wi-fi Password Cracking

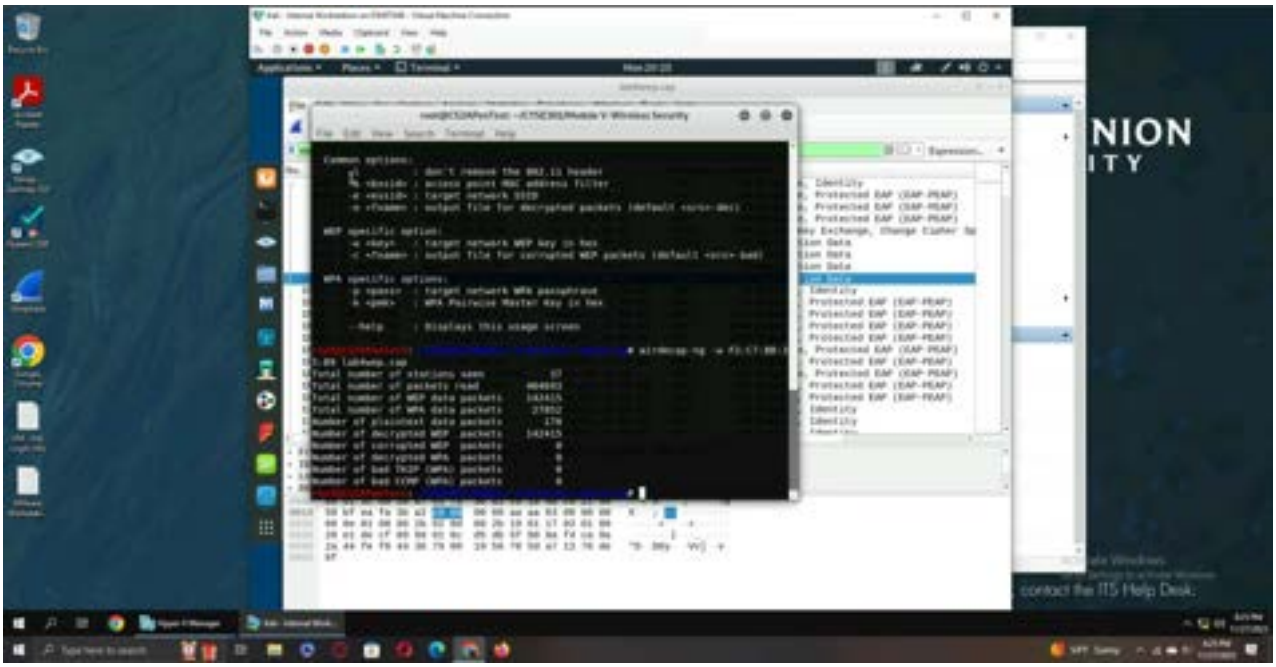
Task A:

Decrypt the lab4wep.cap file:



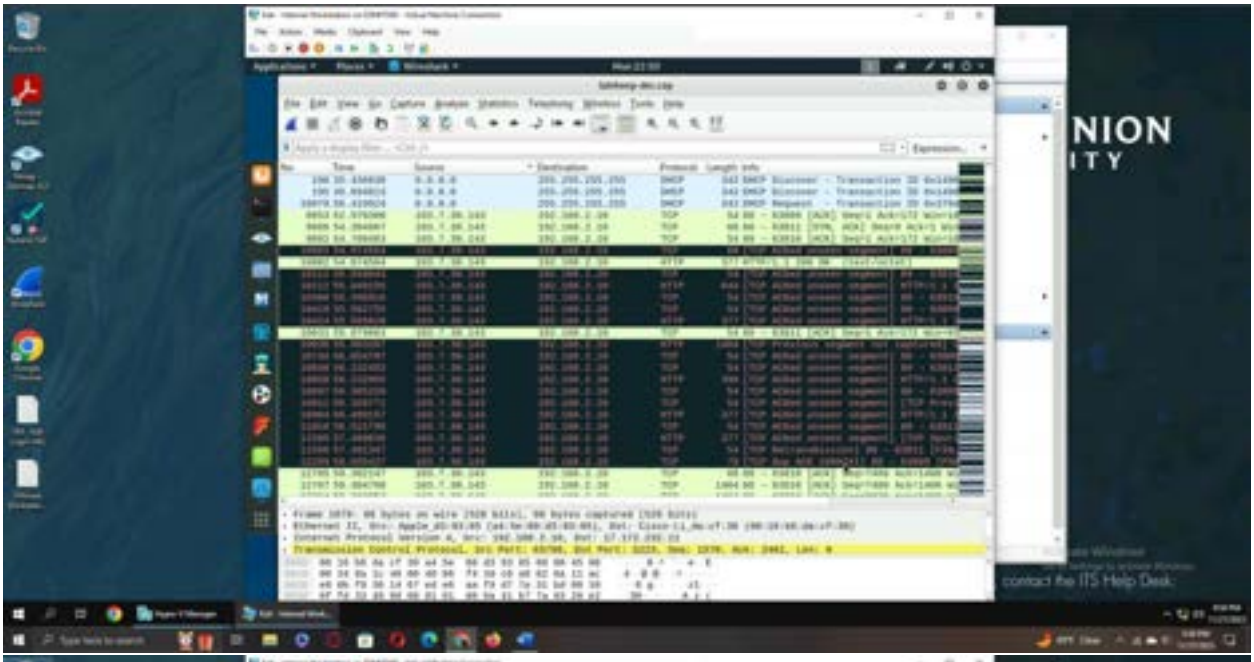


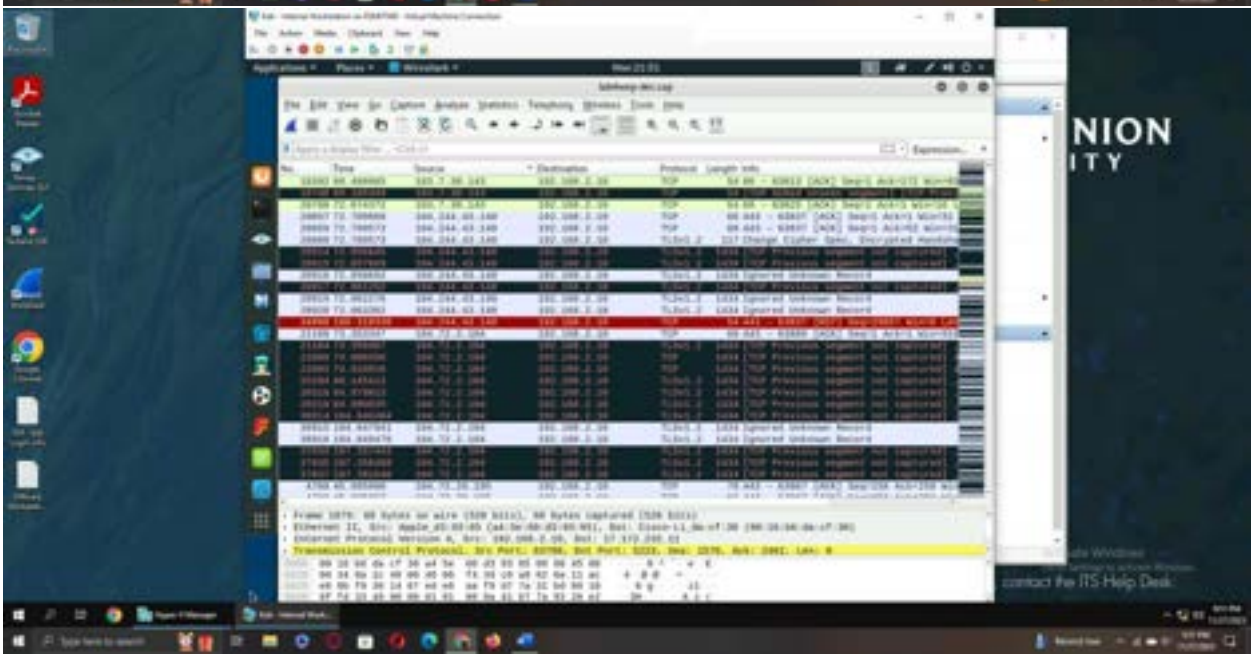


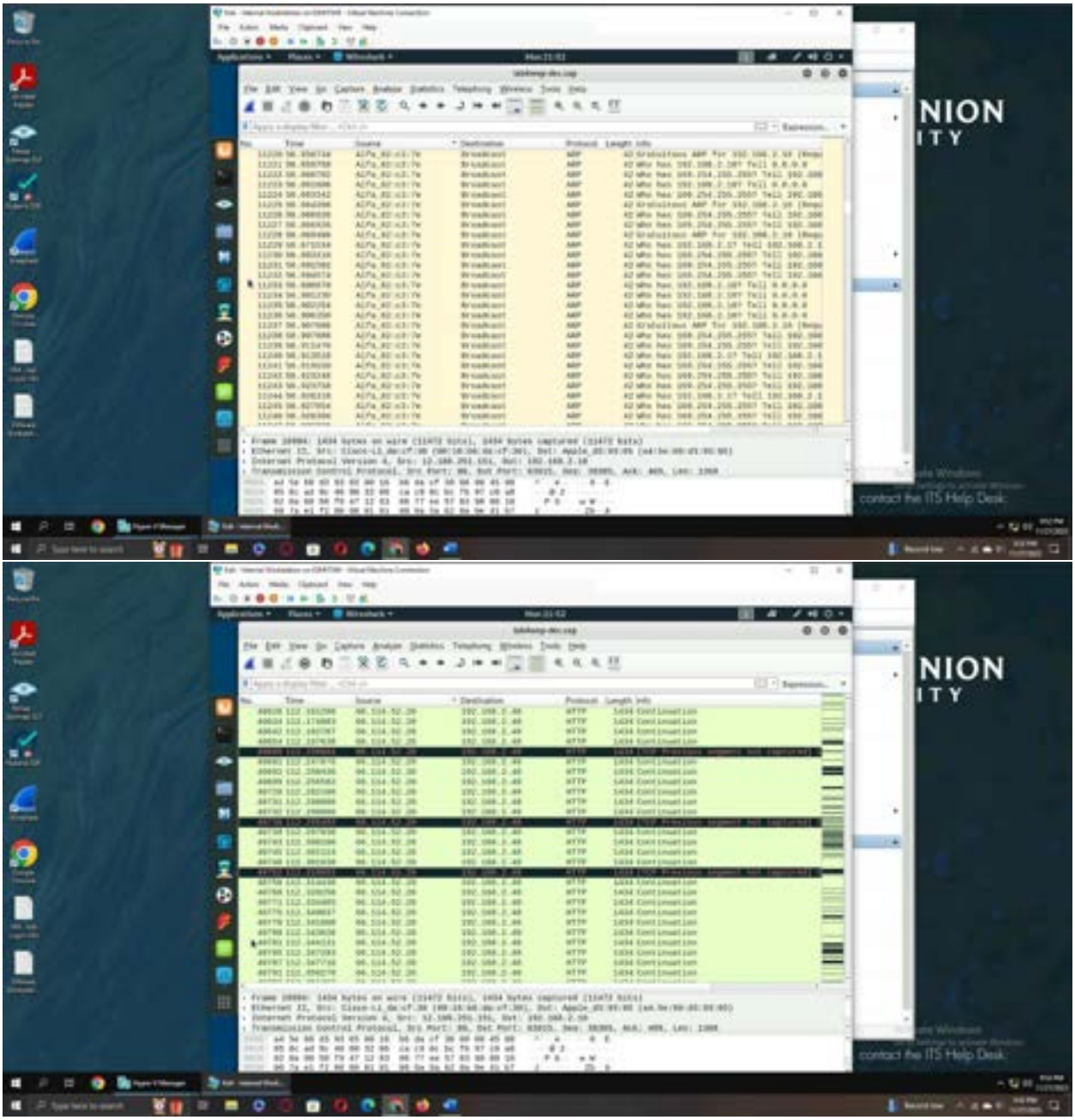


Explanation: Using the aircrack-ng command, I figured out that the first network is the one that needs to be targeted due to it using the WEP. After inputting “1” where it asks for the index, aircrack-ng was able to find the key, which was F2:C7:BB:35:B9. Now that I had the key, I used airdecap-ng, followed by -w along with the key to decrypt the traffic in the lab4wep.cap file. The decrypted traffic was saved into a file named lab4wep-dec.cap, which I then opened in wireshark to analyze the traffic.

Traffic Analysis

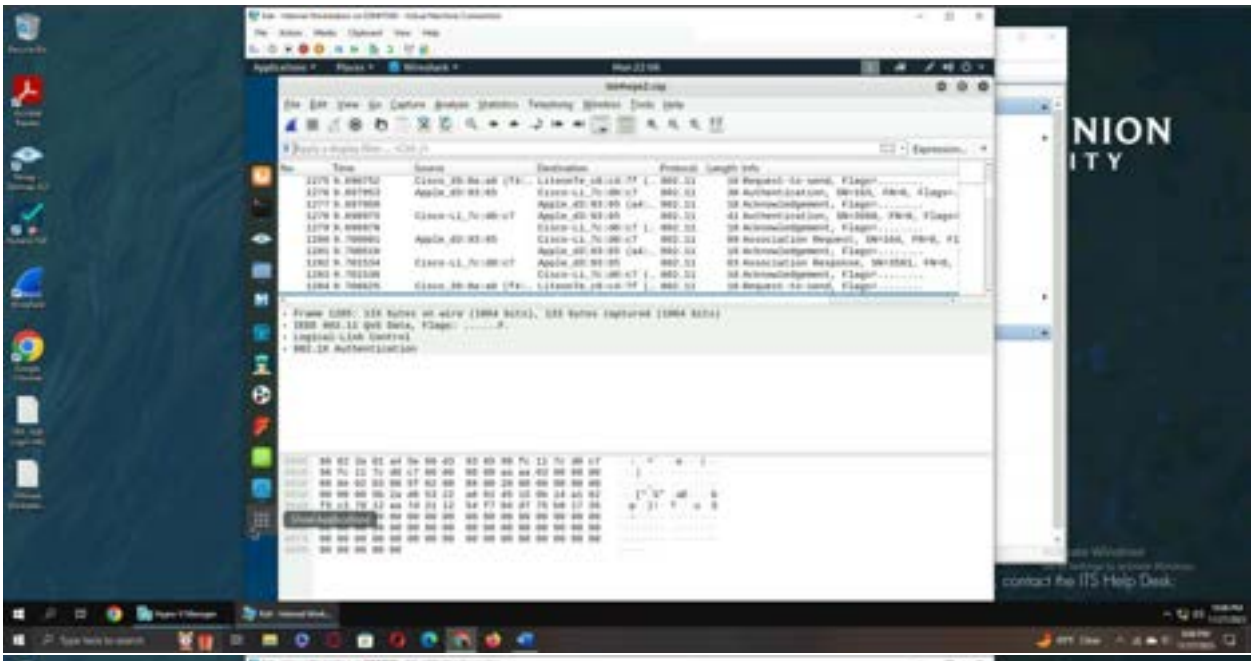

NION
ITY
contact the ITS Help Desk.

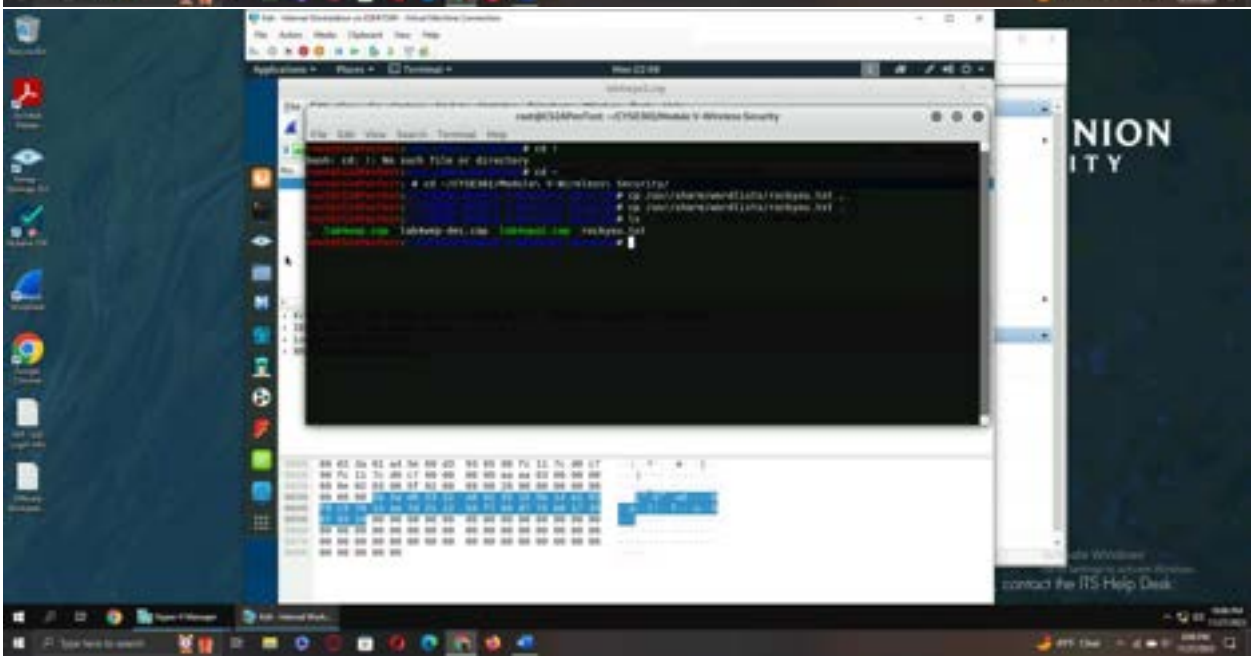

NION
ITY
contact the ITS Help Desk.

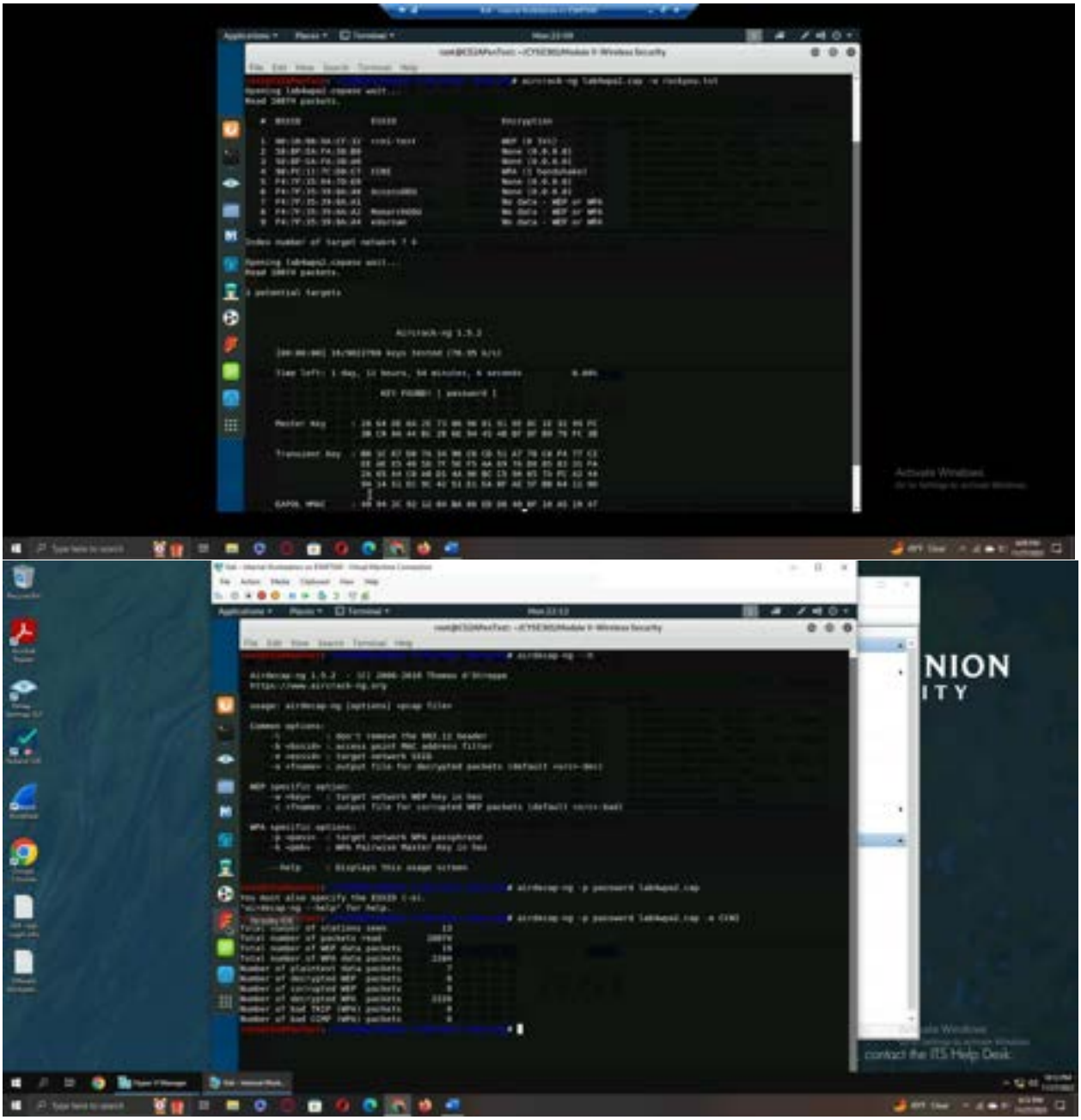


Explanation: The traffic starts off with various TCP SYN packets and HTTP packets being exchanged between various source IP addresses and the destination IP address of 192.168.2.10. There also seem to be a lot of unseen segments, Wireshark most likely couldn't figure out the specifics of the packet but knows it exists and was transferred. However, after a short amount of time, we can see that most of the packets being exchanged over the network are ARP packets, all sharing a source of Alfa_82:c3:7e and a destination of broadcast. The packets seem to be request packets asking about various IP addresses and trying to find their corresponding MAC addresses. While there are some HTTP and TCP packets interspersed between the ARP queries, most of the traffic occurring are ARP query packets.

Decrypt the lab4wpa2.cap file

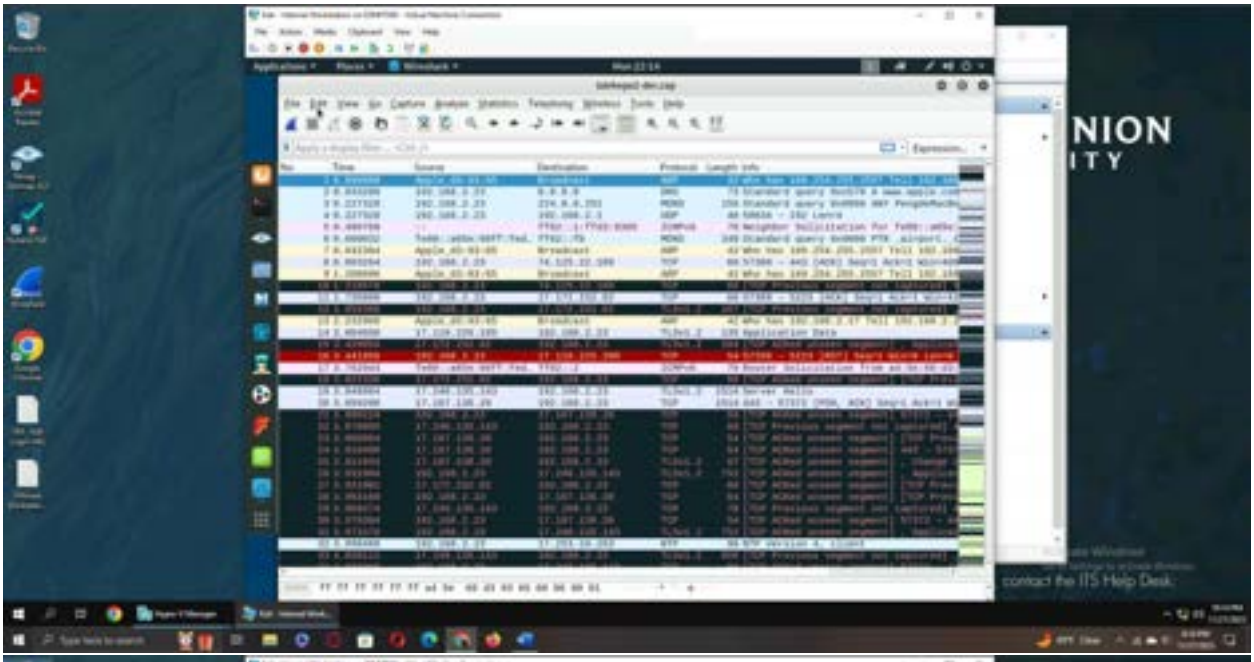


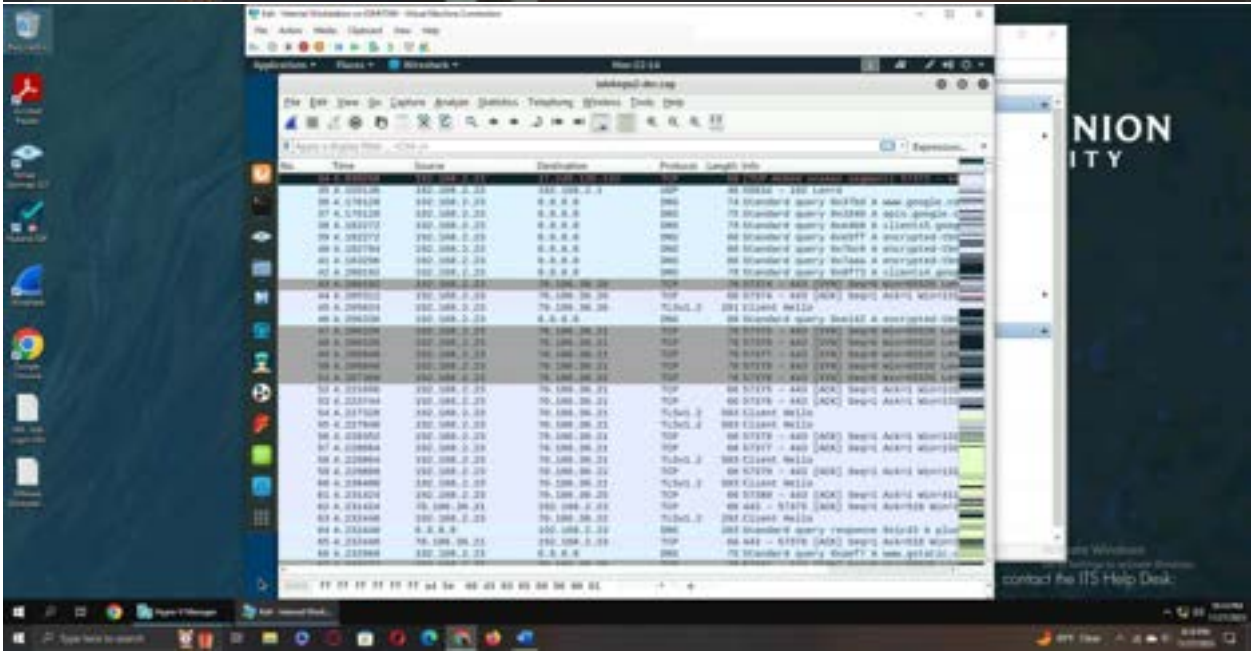


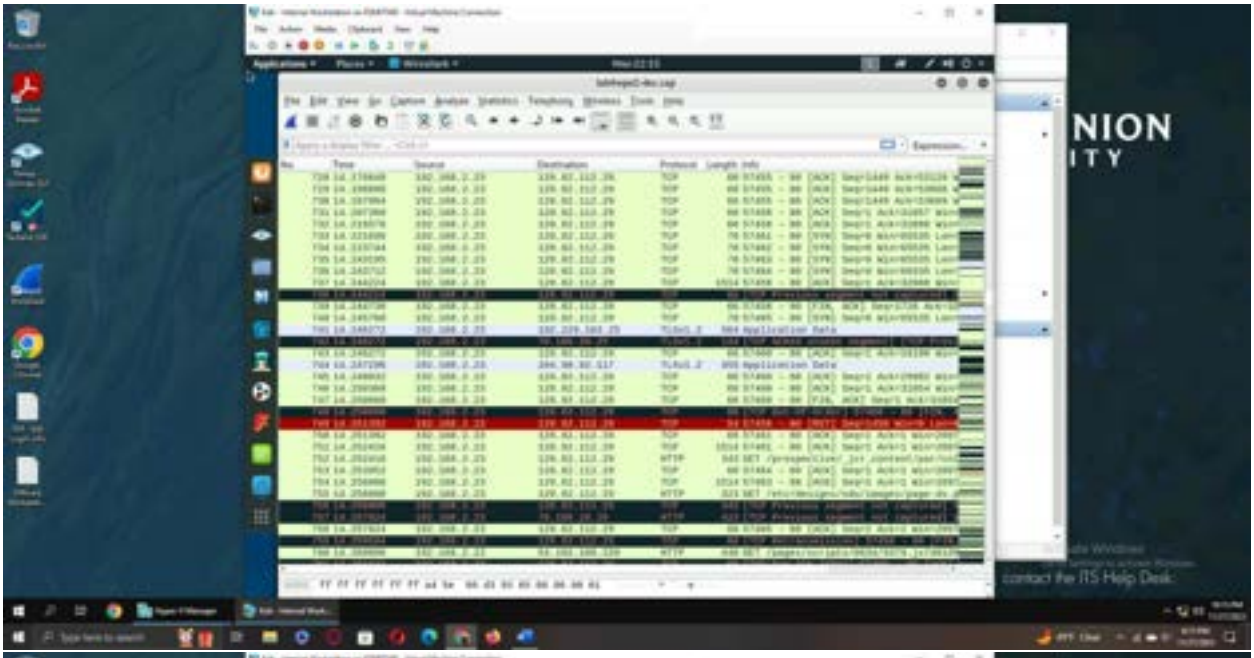


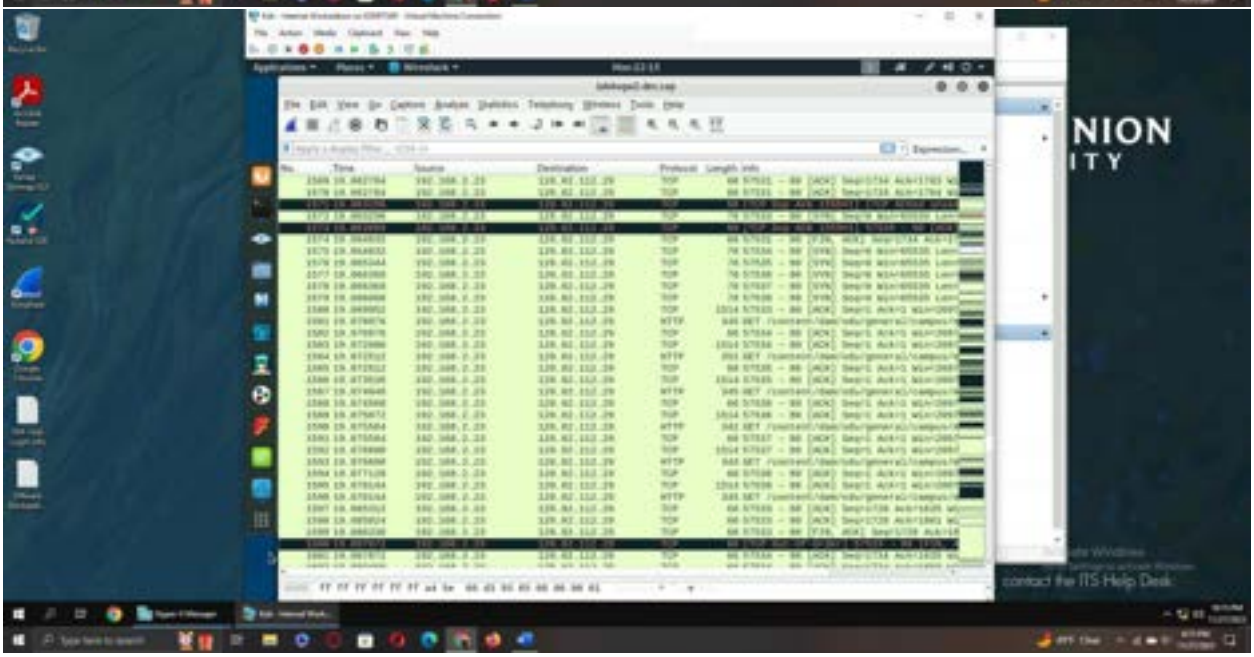
Explanation: After opening the file and examining the encrypted traffic, I copied the rockyou.txt file into my current directory for it to be used in a dictionary attack. After that, I used aircrack-ng to attempt to crack the password using a dictionary attack with the rockyou.txt file as the wordlist. After choosing index 4 from the list of networks, due to it being the wpa format, the dictionary attack occurred, and the “password” key was found. After finding the password, I once use airdecap-ng along with the password and the ESSID of the network, which was CCNI. This allowed me to decrypt the packets, which were placed in a folder called lab4wpa2-dec.cap.

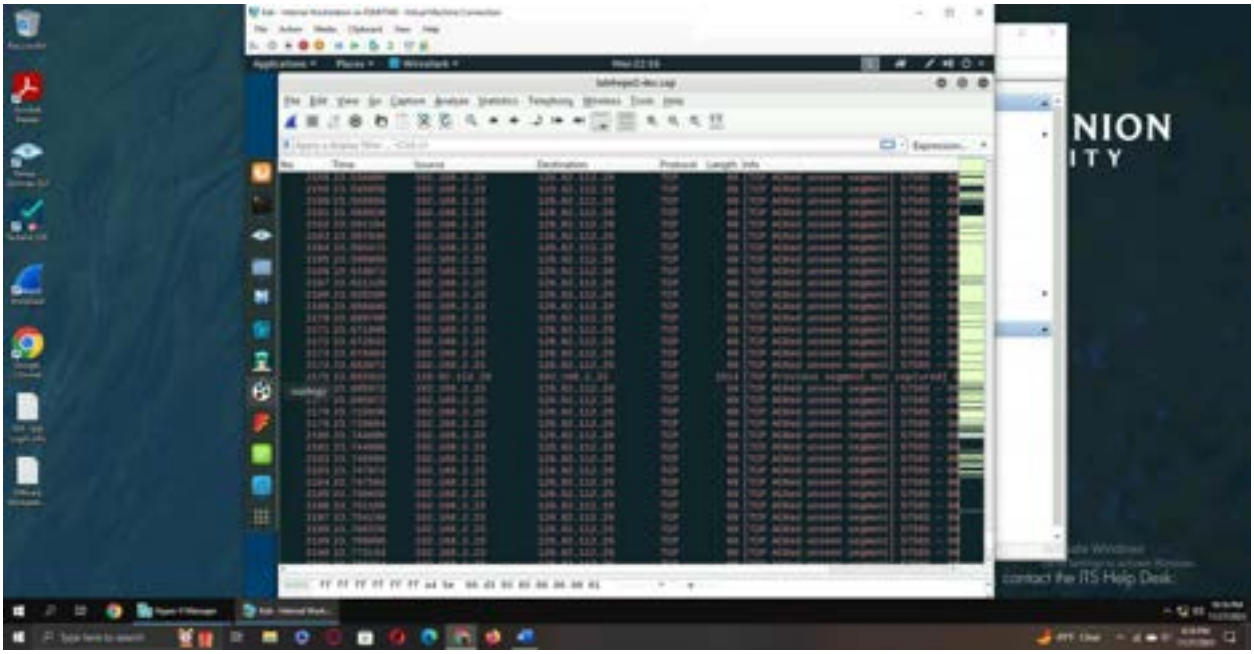
Traffic Analysis


NION
ITY
contact the ITS Help Desk.


NION
ITY
contact the ITS Help Desk.

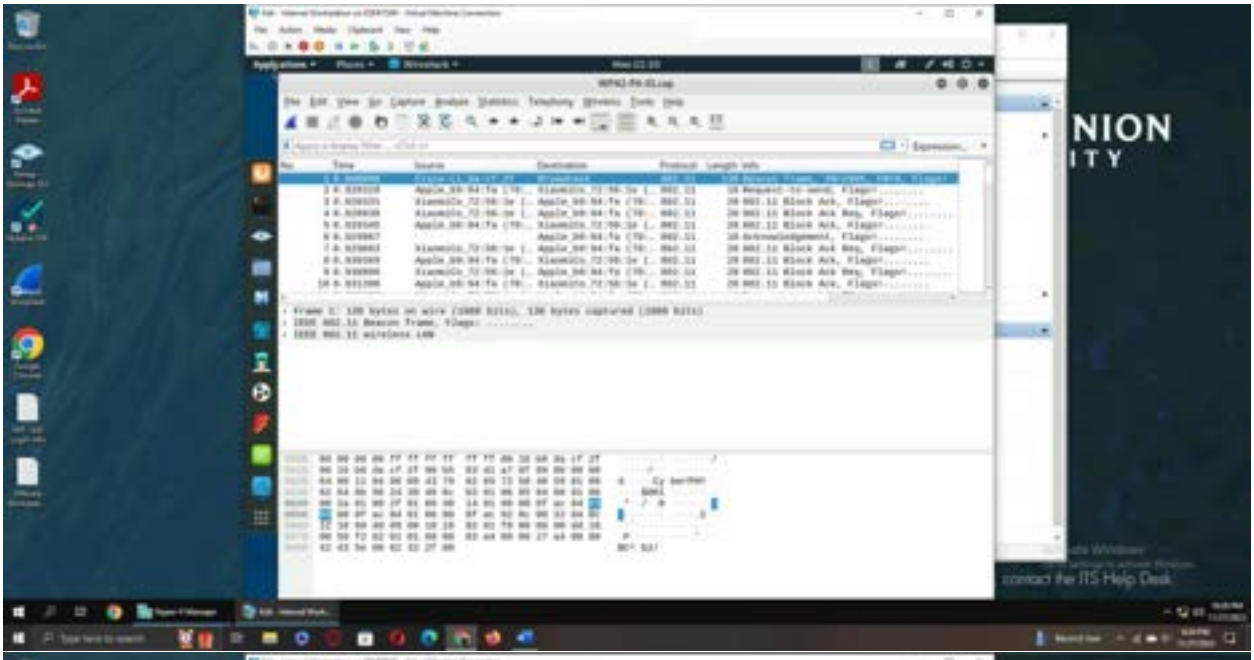

NION
ITY
contact the ITS Help Desk

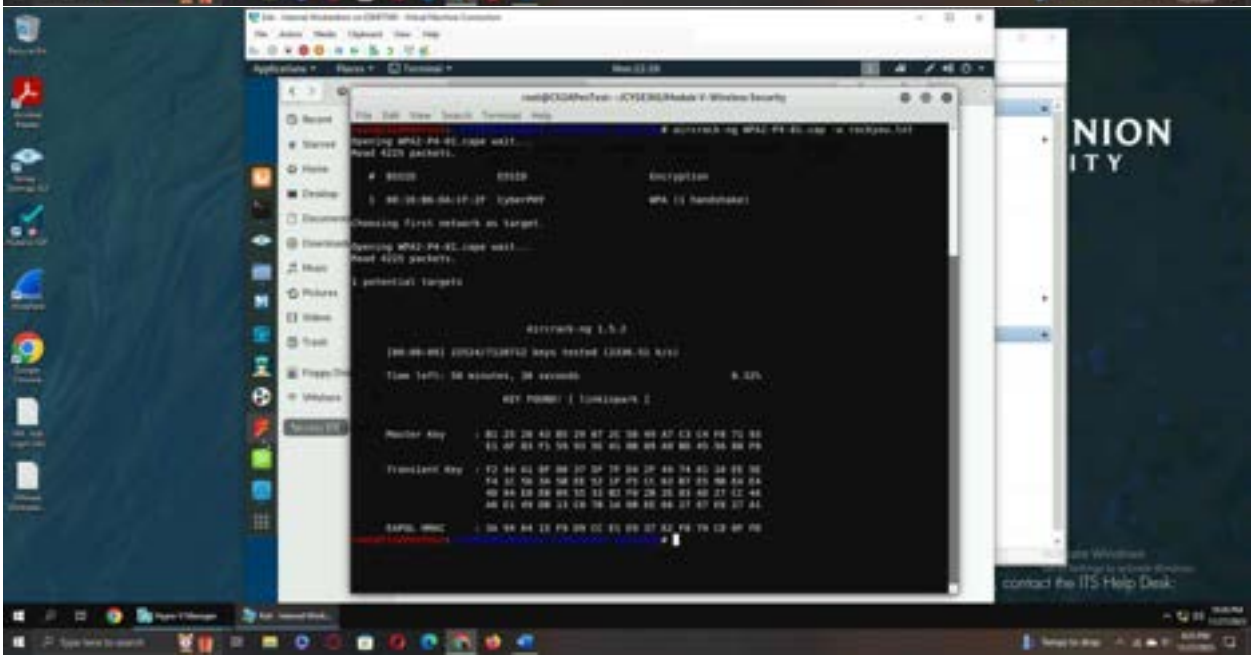

NION
ITY
contact the ITS Help Desk

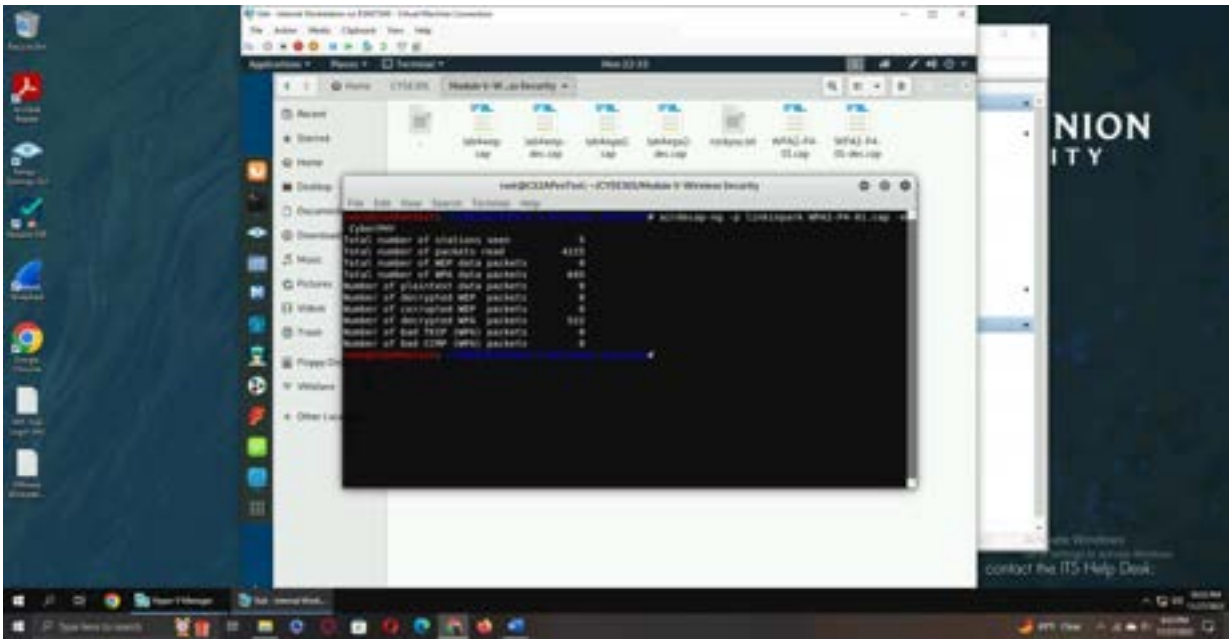


Explanation: A big difference we can immediately notice between the traffic of the wpa2 file and the WEP file is the notable lack of ARP packets in the wpa2 file. While the wep file was mostly ARP query packets, this file has very few to the point where they are basically unnoticeable without filtering the traffic. Much of the traffic occurring over this file are DNS, TCP, and HTTP packets, with some noticeable transfers of UDP and ICMP packets here and there as well. We can also see that there are cases of TLS client hello packets, suggesting that some of the connections have achieved both TCP and TLS handshakes respectively. The lack of ARP packets seems to suggest that the MAC addresses are known this time around, or that they do not need to be found for some reason, but the practical nonexistence of them when compared to the WEP counterpart file is interesting.

Task B

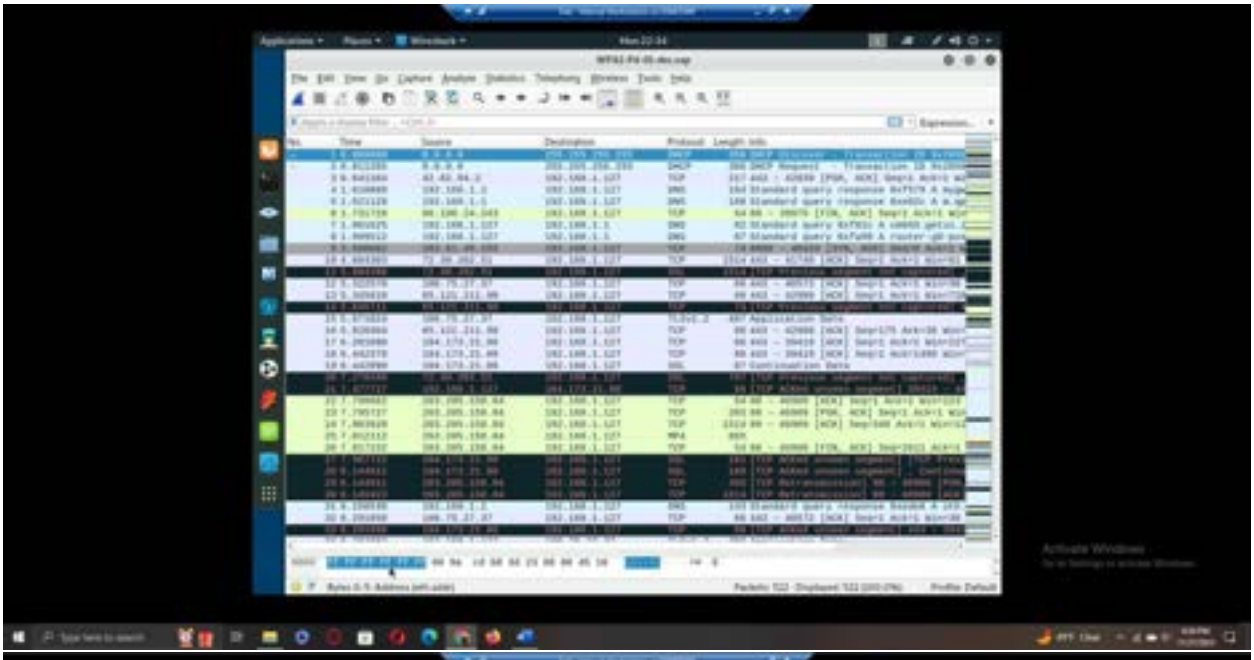


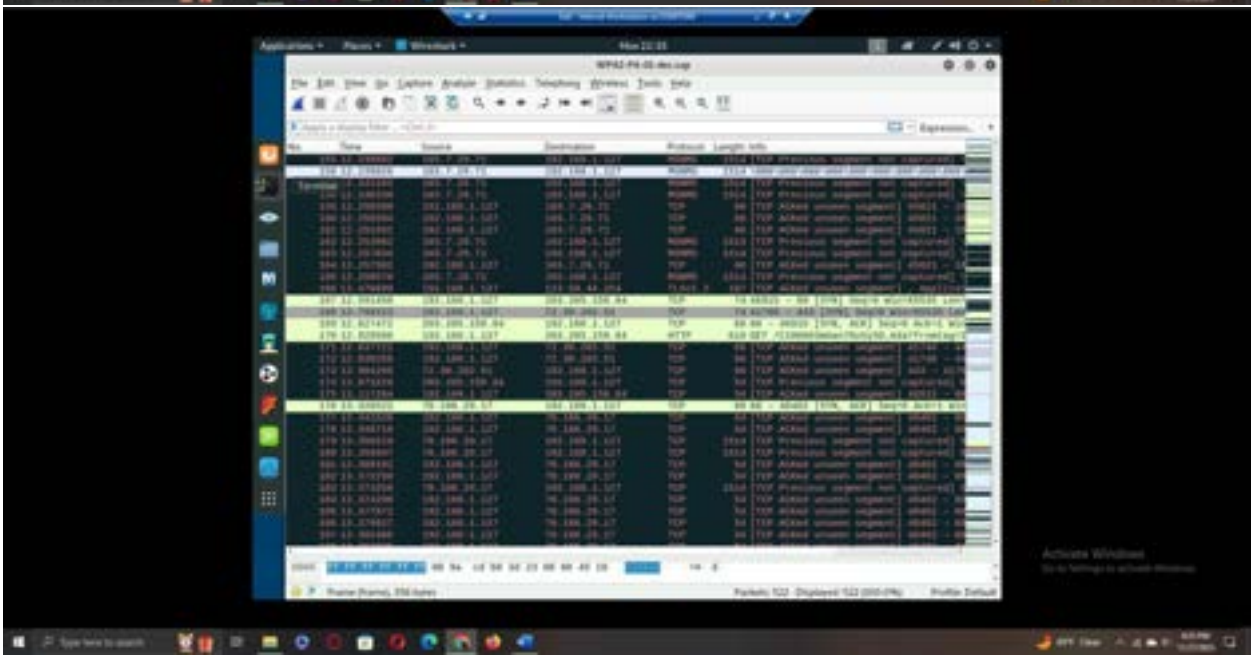


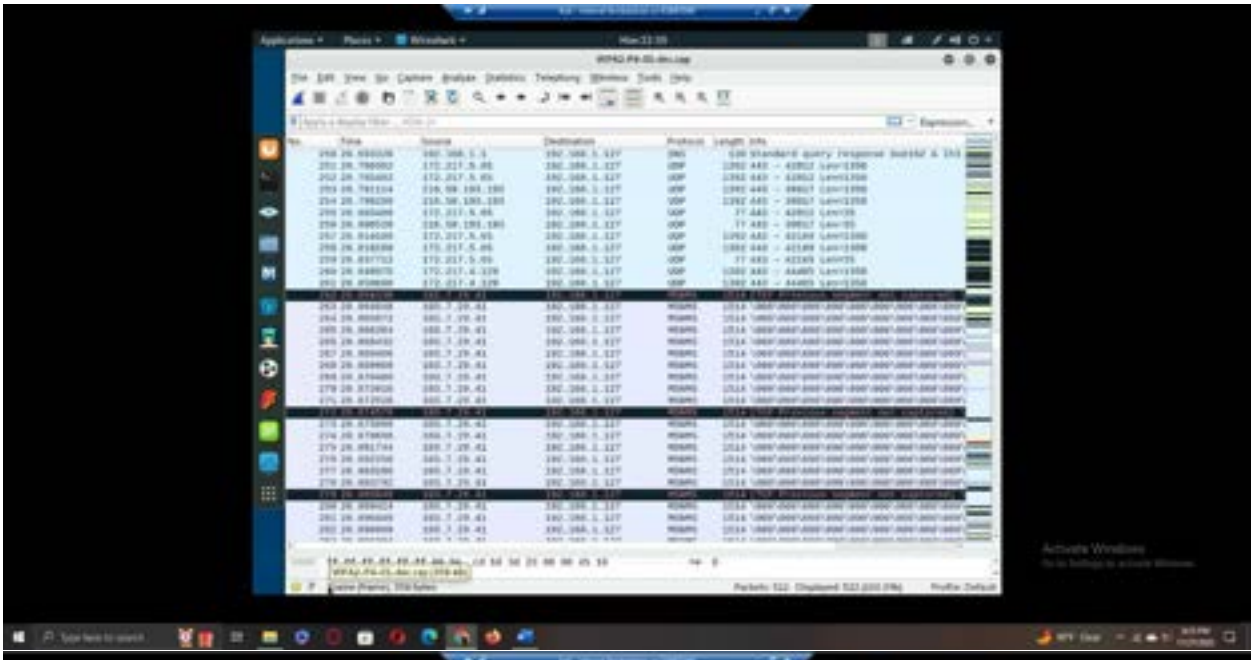


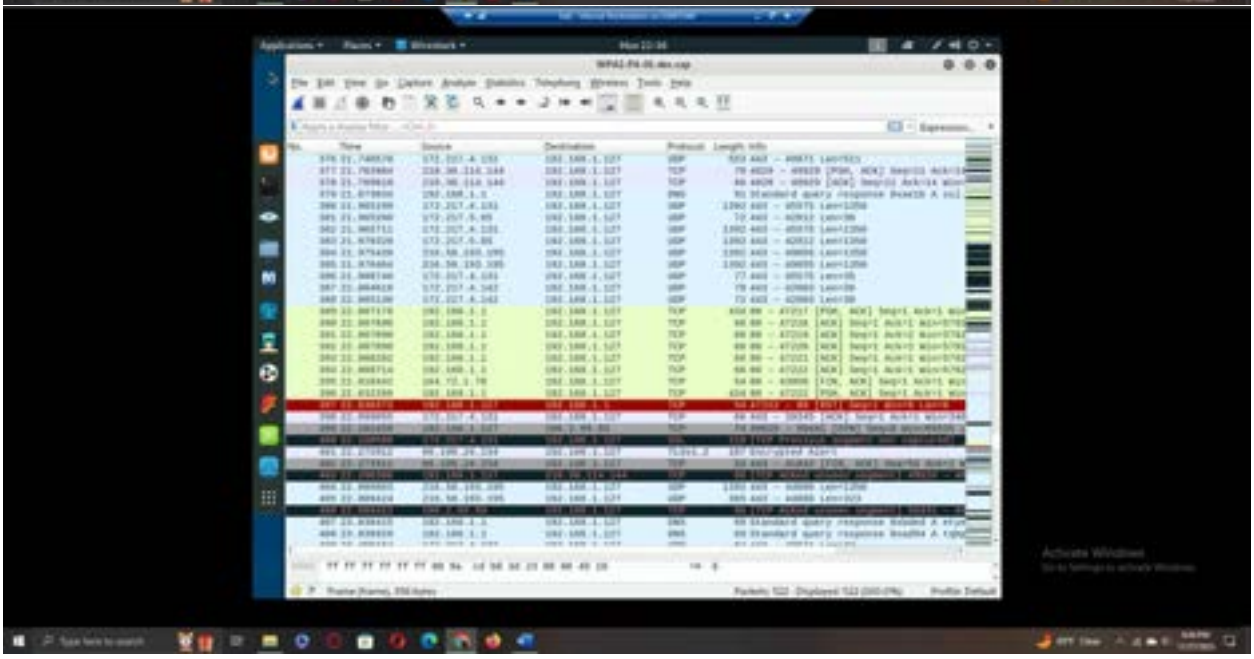
Explanation: After figuring out which file I should use based on my MIDAS ID, I used aircrack-ng to perform a dictionary attack on the WPA2-P4-01.cap file using “rockyou.txt” as the wordlist file. After using that to crack the password (linkinpark), I then used airdecap-ng to decrypt the traffic, using the password I previously cracked along with the ESSID of CyberPHY. This allowed me to decrypt the traffic for analysis, which was saved in a file called WPA2-P4-01-dec.cap

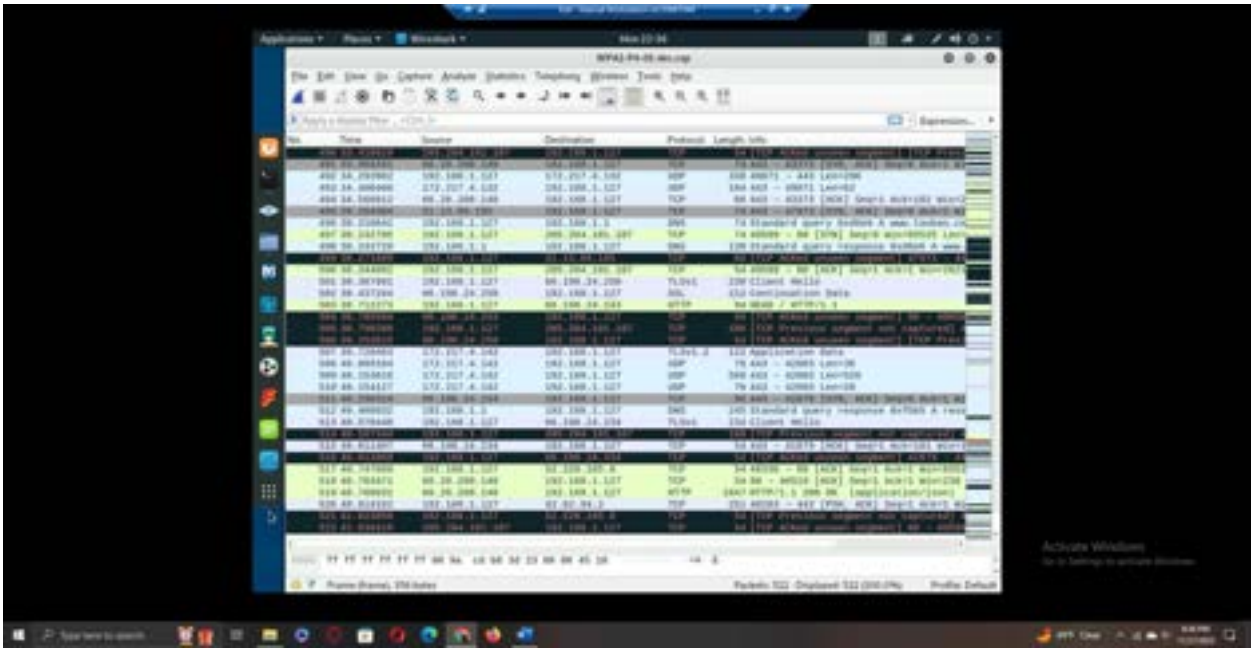
Traffic Analysis











Explanation: We can easily see many similarities between the WPA2-P4-01-dec.cap and lab4wpa2-dec.cap files when analyzing the traffic of both files. When filtering for ARP packets we can once again see that there are none within this traffic file, and most of the file consists of DNS, TCP, and HTTP packets being exchanged. However, this file notably also contains many more instances of UDP packets as well as another packet type called MDNMS which I am not familiar with. Another interesting factor when analyzing this file is that some portions of the analysis contain numerous unseen segments in a row, especially more than the other files. I'm not sure if this is due to the file itself or maybe a malfunction with Wireshark, but it's interesting that this file is the smallest overall yet seems to contain the greatest quantity of unseen packets. This file also has a few examples of TLS client hello packets, again suggesting that some of the connections successfully completed both TCP and TLS handshakes across this wi-fi connection.