Thadomal Shahani Engineering College

Bandra (W.), Mumbai - 400 050.

CERTIFICATE

Certify that Mr./Miss ANIMESH NARAYAN PARAB

of I.T Department, Semester V with

Roll No. 88 has completed a course of the necessary

experiments in the subject SECURITY LAB under my

supervision in the **Thadomal Shahani Engineering College**

Laboratory in the year 2023 - 2024

27/10/23

Teacher In- Charge

Head of the Department

Date 27/10/23

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
1.	Breaking Shift cipher and Mon-alphabetic substitution cipher using Frequency analysis method.		19/7/23	
2	Cryptanalysis or decoding of Polyalphabetic cipher: Playfair, Vigenere cipher.		26/7/23	
3.	Block cipher modes of operation using AES		13/9/23	
4.	Implementation and analysis of RSA cryptosystem and Digital signature scheme using RSA.		26/7/23	
5.	To explore hasdeep tool in kali linux for generating, matching and auditing hash of files.		8/9/23	
6.	Study the use of network reconnaissance tools like WHOIS, dig, traceroute, nslookup, nikto, dmitry to gather info about network and domain registrars.		8/8/23	27/10/23
7.	Study of packet sniffer tools wireshark and TCP DUMP		12/10/12	
8.	Installation of NMAP and using it with different options to scan open ports, perform OS fingerprinting, Ping Scan, TCP Port Scan, UDP Port Scan etc.		8/8/23	
9.	Simulate DOS attack using Hping3		9/8/23	
10.	To study and Configure Firewalls using IP tables		12/9/23	
11.	Installing Snort, setting in Intrusion Detection Mode and writing rules for Intrusion detection.		13/9/23	
12.	Explore the GnPG tools of linux to implement email security.		21/9/23	
13.	Written Assignment 1		21/9/23	
14.	written Assignment 2			

Assignment 1

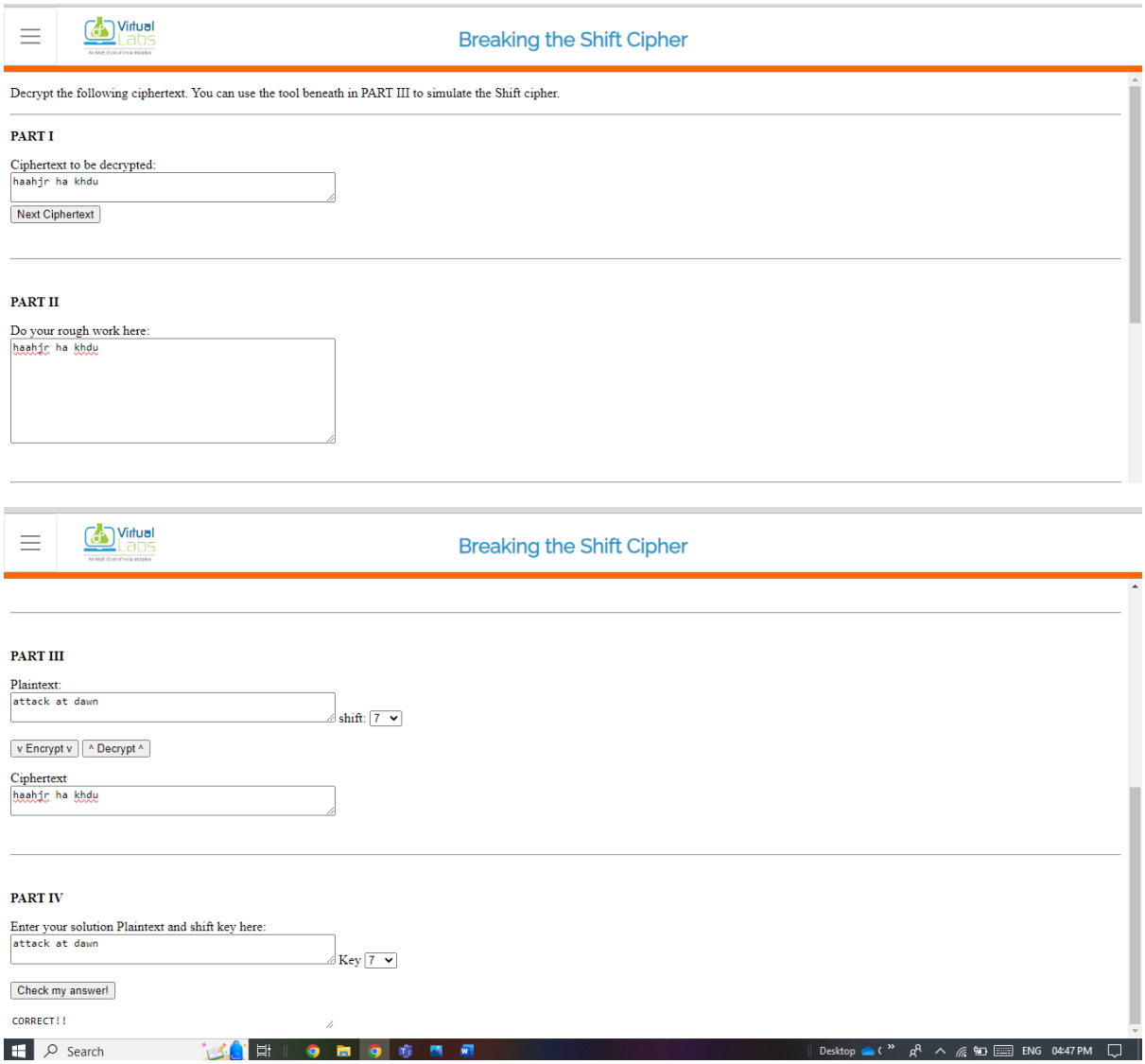
Aim: Breaking shift cipher and Mono-alphabetic Substitution Cipher using Frequency analysis method.

Lab outcome: Illustrate symmetric cryptography by implementing classical ciphers

Theory:

What is shift cipher?

A shift cipher, also known as a Caesar cipher, is a basic and straightforward encryption technique used in cryptography. It is one of the earliest and simplest methods of encryption. In a shift cipher, each letter in the plaintext is shifted a fixed number of positions down or up the alphabet. The number of positions shifted is called the "key" or "shift value." For example, with a shift of 3, "A" would be encrypted as "D," "B" would become "E," and so on. The shift wraps around the alphabet, so "Z" would be encrypted as "C' with a shift of 3.

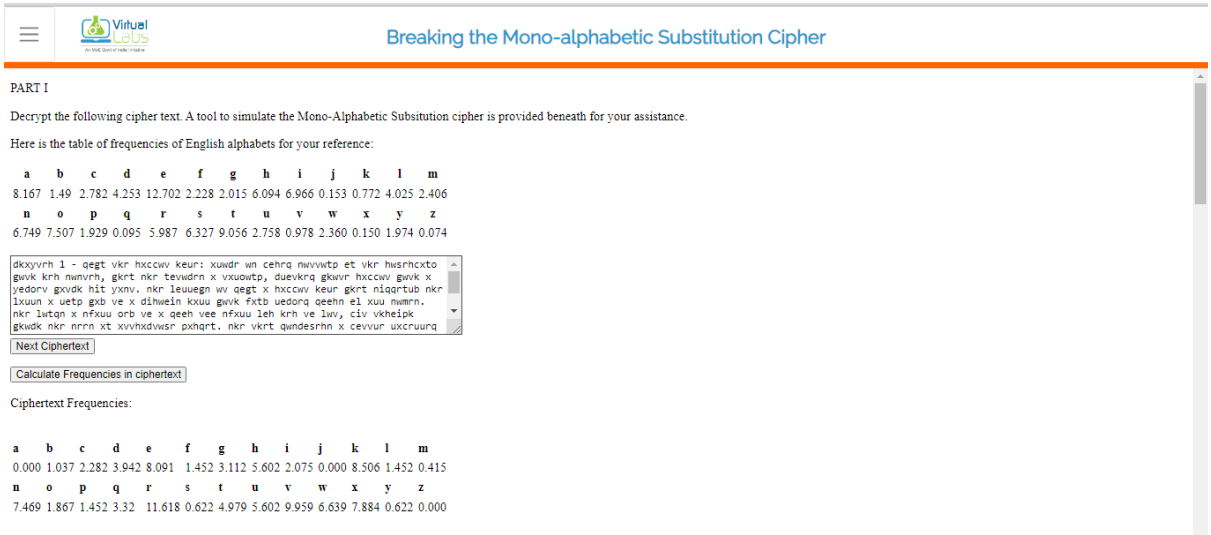


How and why it can be broken using brute force attack?

A shift cipher (Caesar cipher) can be broken using a brute force attack due to its limited key space. Since there are only 26 possible keys in the case of the English alphabet (shifting 0 to 25 positions), an attacker can systematically try all possible keys to decrypt the ciphertext. Here's how a brute force attack works on a shift cipher: Enumerate all possible keys: Since there are only 26 possible keys, the attacker tries all combinations, shifting the ciphertext by 0 to 25 positions. Decrypt the ciphertext: For each key, the attacker shifts the ciphertext letters back by the corresponding number of positions to recover the plaintext. Check for meaningful results: After decrypting with each key, the attacker examines the output to see if it resembles meaningful English text. The assumption is that the correct key will produce a message that is easily recognizable as a coherent sentence or phrase. Verify the result: The attacker usually employs some form of automated or manual analysis to check the output for linguistic patterns, common words, or other recognizable features that indicate a successful decryption. Select the correct key: Once the attacker finds the decryption that appears to be meaningful English text, they have likely found the correct key, and the message is deciphered. The reason a shift cipher is vulnerable to brute force attacks is that it has a very small key space. With only 26 possible keys, trying all combinations is a relatively trivial task for a computer program or a determined attacker. This simplicity is why the shift cipher is considered weak and is not used for secure encryption purposes today.

What is monoalphabetic cipher?

A monoalphabetic cipher is a type of substitution cipher where each letter in the plaintext is replaced with a fixed corresponding letter in the ciphertext. In other words, the same substitution rule is applied consistently throughout the entire message. This means that each occurrence of a specific letter in the plaintext will always be replaced by the same letter in the ciphertext



PART II

Note that the *cipher text is in lower case* and when you replace any character, the final character of replacement, i.e., *plaintext is changed to upper case* automatically in the following scratchpad.

Scratchpad:

```
CHAPTER 1 - DOWN THE RABBIT HOLE: ALICE IS BORED SITTING ON THE RIVERBANK
WITH HER SISTER, WHEN SHE NOTICES A TALKING, CLOTHED WHITE RABBIT WITH A
POCKET WATCH RUN PAST. SHE FOLLOWS IT DOWN A RABBIT HOLE WHEN SUDDENLY SHE
FALLS A LONG WAY TO A CURIOUS HALL WITH MANY LOCKED DOORS OF ALL SIZES. SHE
FINDS A SMALL KEY TO A DOOR TOO SMALL FOR HER TO FIT, BUT THROUGH WHICH SHE
SEES AN ATTRACTIVE GARDEN. SHE THEN DISCOVERS A BOTTLE LABELLED 'DRINK ME',
THE CONTENTS OF WHICH CAUSE HER TO SHRINK TOO SMALL TO REACH THE KEY. A
CAKE WITH 'EAT ME' ON IT CAUSES HER TO GROW TO SUCH A TREMENDOUS SIZE HER
HEAD HITS THE CEILING.
```

Modify the text above (in scratchpad):

This is case sensitive function and replaces only cipher text (lower case) by plain text (upper case):

Replace cipher character [m] by plaintext character [Z] Modify

Use the following function to undo any unwanted exchange by giving an uppercase character and a lower case. This is a case sensitive function:

Replace character [ ] by character [ ] Replace these exact characters

Your replacement history:

You replaced d by C You replaced k by H You replaced x by A You replaced y by P You replaced v by T You replaced r by E You replaced h by R You replaced q by D You replaced e by O You replaced g by W You replaced t by N You replaced c by B You replaced w by I You replaced u by L You replaced n by S You replaced p by G You replaced s by V You replaced o by K You replaced i by U You replaced l by F You replaced b by Y You replaced f by M You replaced m by Z

Virtual Labs

Breaking the Mono-alphabetic Substitution Cipher

PART III

Enter your solution plaintext here:

```
CHAPTER 1 - DOWN THE RABBIT HOLE: ALICE IS BORED SITTING ON THE
RIVERBANK WITH HER SISTER, WHEN SHE NOTICES A TALKING, CLOTHED WHITE
RABBIT WITH A POCKET WATCH RUN PAST. SHE FOLLOWS IT DOWN A RABBIT HOLE
WHEN SUDDENLY SHE FALLS A LONG WAY TO A CURIOUS HALL WITH MANY LOCKED
DOORS OF ALL SIZES. SHE FINDS A SMALL KEY TO A DOOR TOO SMALL FOR HER
```

Solution Key = xcdqrlpkwzoufteyahnvisgjbm

Check Answer!

CORRECT!!

How and why it can be broken using brute force attack?

A shift cipher (Caesar cipher) can be broken using a brute force attack due to its limited key space. Since there are only 26 possible keys in the case of the English alphabet (shifting 0 to 25 positions), an attacker can systematically try all possible keys to decrypt the ciphertext. Here's how a brute force attack works on a shift cipher: Enumerate all possible keys: Since there are only 26 possible keys, the attacker tries all combinations, shifting the ciphertext by 0 to 25 positions. Decrypt the ciphertext: For each key, the attacker shifts the ciphertext letters back by the corresponding number of positions to recover the plaintext. Check for meaningful results: After decrypting with each key, the attacker examines the output to see if it resembles meaningful English text. The assumption is that the correct key will produce a message that is easily recognizable as a coherent sentence or phrase.

Verify the result: The attacker usually employs some form of automated or manual analysis to check the output for linguistic patterns, common words, or other recognizable features that indicate a successful decryption. Select the correct key: Once the attacker finds the decryption that appears to be meaningful English text, they have likely found the correct key, and the message is deciphered.

The reason a shift cipher is vulnerable to brute force attacks is that it has a very small key space. With only 26 possible keys, trying all combinations is a relatively trivial task for a

computer program or a determined attacker. This simplicity is why the shift cipher is considered weak and is not used for secure encryption purposes today.

How it is broken using frequency analysis attack?

Breaking the Caesar cipher is trivial as it is vulnerable to most forms of attack. The system is so easily broken that it is often faster to perform a brute force attack to discover if this cipher is in use or not. An easy way for humans to decipher it is to examine the letter frequencies of the cipher text and see where they match those found in the underlying language.

By graphing the frequencies of letters in the ciphertext and those in the original language of the plaintext, a human can spot the value of the key by looking at the displacement of particular features of the graph. For example in the English language the frequencies of the letters Q,R,S,T have a particularly distinctive pattern.

Conclusion :

We understood the working of shift and monoalphabetic cipher and successfully implemented the simulation of shift and monoalphabetic cipher using virtual lab

Assignment 2

Aim:- Cryptanalysis or decoding of polyalphabetic ciphers: Playfair, Vigenere cipher.

Lo Mapped:- Lo 1

Theory:-

Playfair Cipher:

• The Playfair Cipher is a polygraphic substitution cipher that operates on pairs of letters (digraphs) instead of individual letters.

• It uses a 5x5 matrix (usually called a key square) containing a keyword's unique letters followed by the remaining letters of the alphabet (excluding duplicates).

• The matrix is used to encrypt and decrypt letters in pairs.

• Encryption Steps:

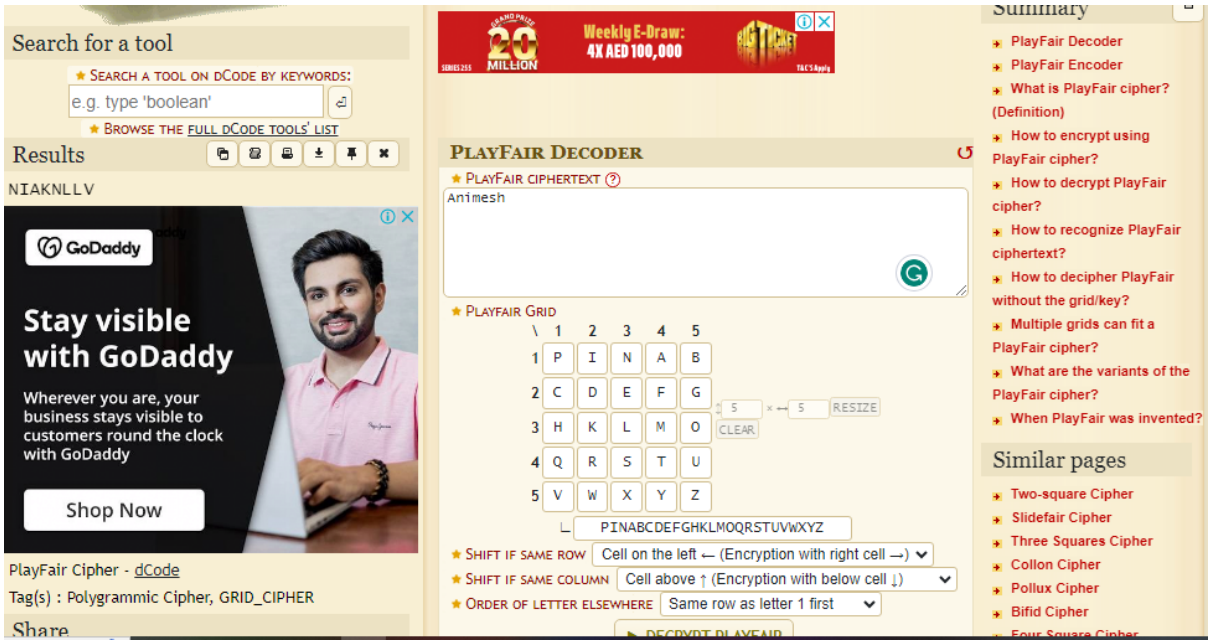
1. Break the plaintext into pairs (digraphs) of letters.

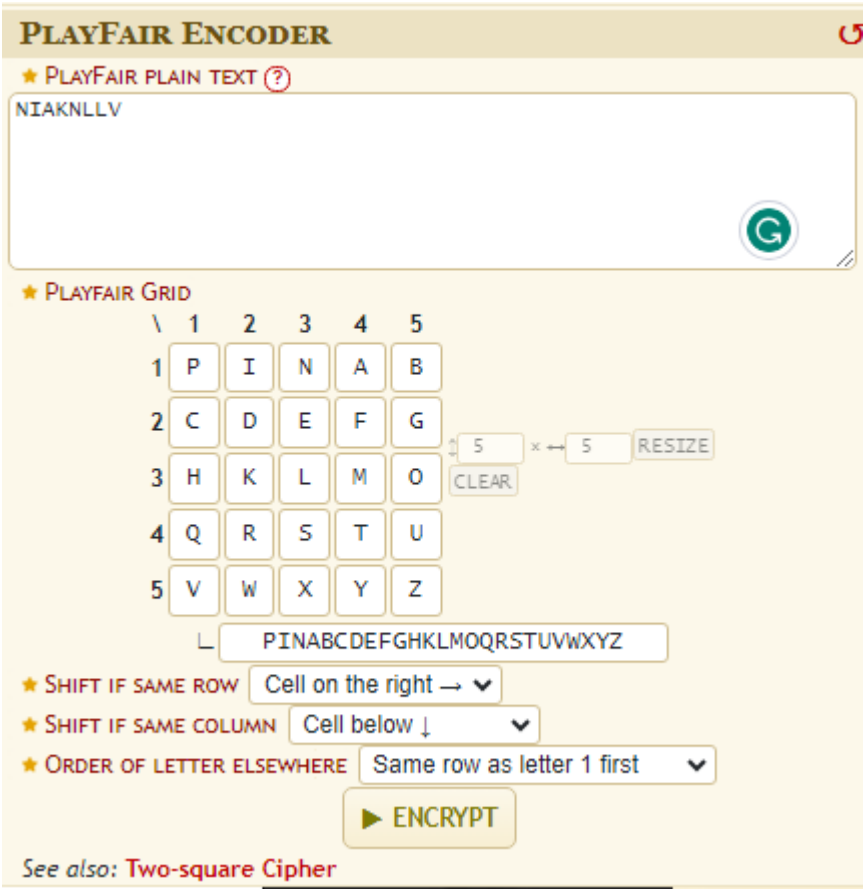
2. If a pair has identical letters, insert a filler letter (like X) between them.

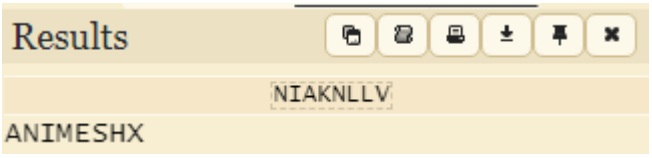
3. For each digraph, find the positions of its letters in the key square.

4. Apply specific rules to determine the ciphertext letters based on the positions.

• Decryption is essentially the reverse process of encryption.







Vigenère Cipher:

• The Vigenère Cipher is a polyalphabetic substitution cipher that uses a keyword to determine different shift values for each letter.

• It's an improvement over the Caesar Cipher, where each letter can be shifted by a different amount.

• Encryption Steps:

1. Repeatedly write the keyword above the plaintext.

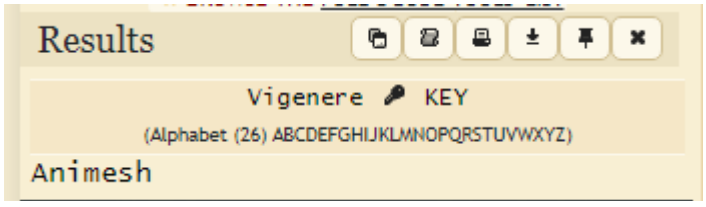
2. Convert both the keyword and the plaintext into numbers using a key-to-number mapping.

3. Add the numbers of the keyword and the plaintext modulo the size of the alphabet to get the ciphertext numbers.

4. Convert the ciphertext numbers back to letters using a number-to-key mapping.

• The keyword's length determines the periodicity of the cipher's key. Longer keywords increase security.

• Decryption requires subtracting the keyword values from the ciphertext values and then converting back to plaintext letters.



Conclusion:
We conclude that both the Playfair Cipher and the Vigenère Cipher offer improvements over basic substitution ciphers, adding complexity and making frequency analysis more difficult. However, they can still be vulnerable to more sophisticated attacks, and modern cryptographic methods like the Advanced Encryption Standard (AES) have replaced them in secure communication.

Animesh Parab T2-T21 88

ASSIGNMENT NO – 3

AIM: To understand AES cipher with various block cipher modes

LAB OUTCOME: LO2: Demonstrate Key Management, Distribution and Authentication.

THEORY:

1. **Briefly explain AES algorithm (What type of cipher it is?, number of rounds, keysize, block size, operations in each round)**

☐ **AES (Advanced Encryption Standard)** is a symmetric-key block cipher that is widely used for secure data encryption. Here's a brief overview of its key characteristics along with an example:

1)Cipher Type: AES is a symmetric-key cipher, meaning the same key is used for both encryption and decryption.

2)Number of Rounds: The number of rounds in AES depends on the key size:

AES-128: 10 rounds

AES-192: 12 rounds

AES-256: 14 rounds

3)Key Size: AES supports three different key sizes: 128 bits (16 bytes), 192 bits (24 bytes), and 256 bits (32 bytes).

4)Block Size: AES processes data in fixed-size blocks of 128 bits (16 bytes). This means that data to be encrypted is divided into blocks, and each block is processed separately.

5)Operations in Each Round: Each round of AES consists of several operations:

a. **SubBytes:** Substitutes each byte in the block with a corresponding byte from a fixed S-box.

b. **ShiftRows:** Rearranges the bytes within each row of the block.

c. **MixColumns:** Mixes the columns within the block using a mathematical transformation.

d. **AddRoundKey:** XORs the block with a round-specific subkey derived from the original encryption key.

Example:

Let's encrypt a simple message "AES" using AES-128 with the key "EXAMPLEKEY12345." We'll perform one round of AES encryption:

Message: "AES"

Key: "EXAMPLEKEY12345" (128 bits)

Initial Round:

a. SubBytes: Replace each byte with a corresponding byte from the S-box.

Input: "AES"

Output (after substitution): "BDFA92"

b. ShiftRows: Rearrange bytes within rows (no change in the initial round).

c. MixColumns: Mix the columns within the block (not done in the initial round).

d. AddRoundKey: XOR the block with the first round key derived from the original key.

Initial Key: "EXAMPLEKEY12345"

Round 1 Key (derived from initial key): "375920614F761A3ECF4C1106D991F37F"

Input: "BDFA92"

Output (after XOR): "8863F2"

The encrypted message after one round is "8863F2." The remaining rounds are applied (9 more rounds for AES-128) to complete the encryption process.Decrypting the message involves reversing these steps with the same key and round keys in reverse order.

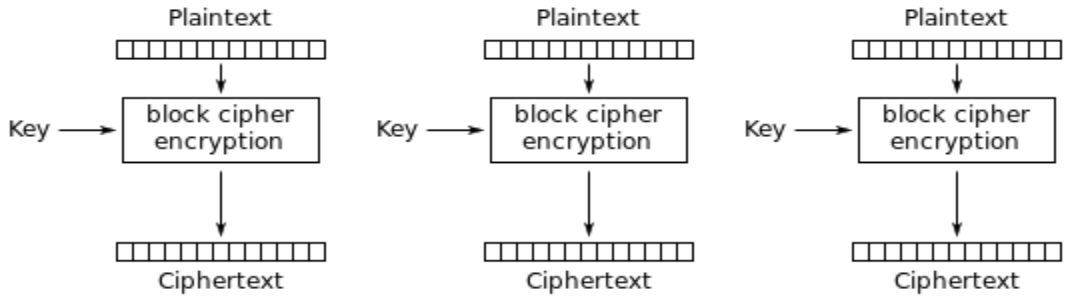
AES is a widely recognized and secure encryption standard used for protecting sensitive data in various applications, including secure communication, data storage, and more.

2. With diagram explain in brief block cipher modes of operation

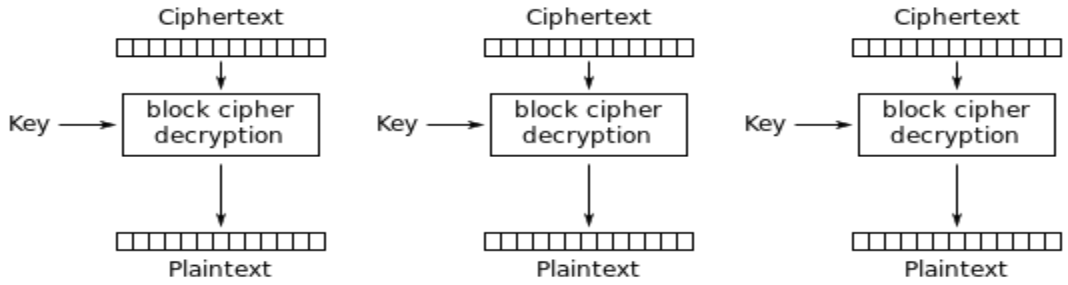
☐ Encryption algorithms are divided into two categories based on the input type, as a block cipher and stream cipher.

a)Electronic Code Book (ECB):–

Electronic code book is the easiest block cipher mode of functioning. It is easier because of direct encryption of each block of input plaintext and output is in form of blocks of encrypted ciphertext. Generally, if a message is larger than b bits in size, it can be broken down into a bunch of blocks and the procedure is repeated.



Electronic Codebook (ECB) mode encryption



Electronic Codebook (ECB) mode decryption

Advantages of using ECB:–

- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption.
- Simple way of the block cipher.

Disadvantages of using ECB:–

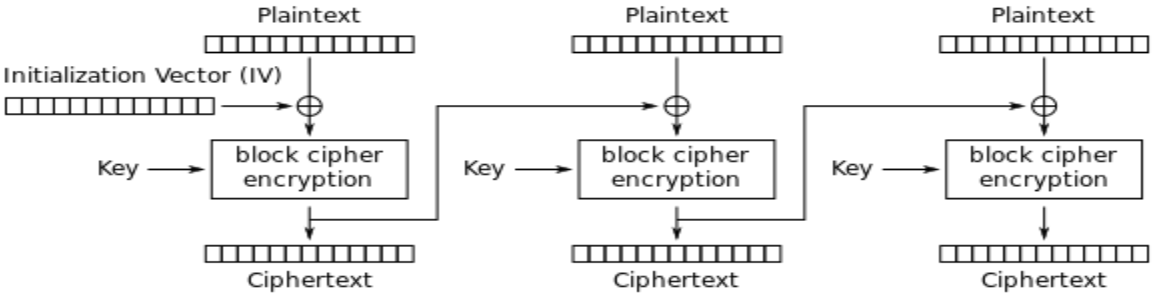
☐ Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertext.

b)Cipher Block Chaining(CBC):–

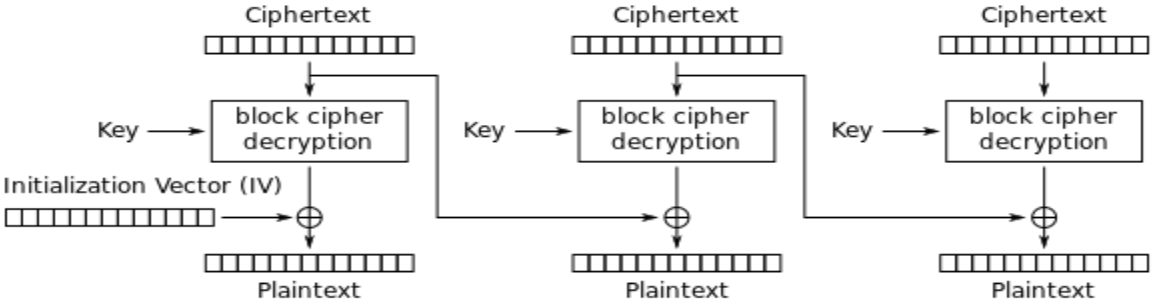
Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC, the previous cipher block is given as input to the

next encryption algorithm after XOR with the original plaintext block. In a nutshell here, a cipher block is produced by encrypting an XOR output of the previous cipher block and present plaintext block.

The process is illustrated here:



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Advantages of CBC:–

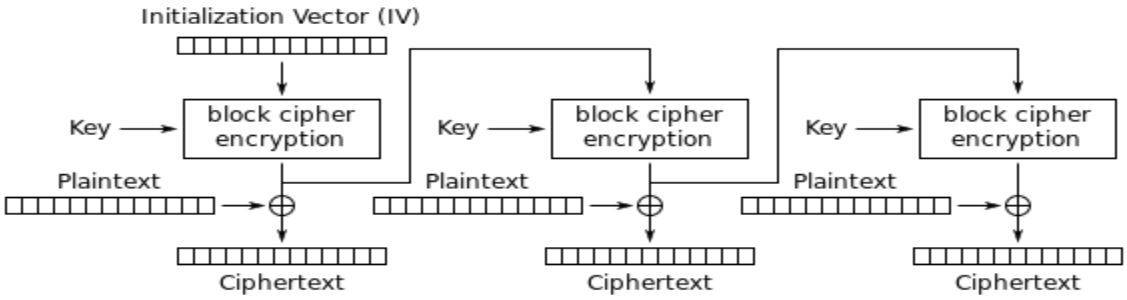
- CBC works well for input greater than b bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalysis than ECB.

Disadvantages of CBC:–

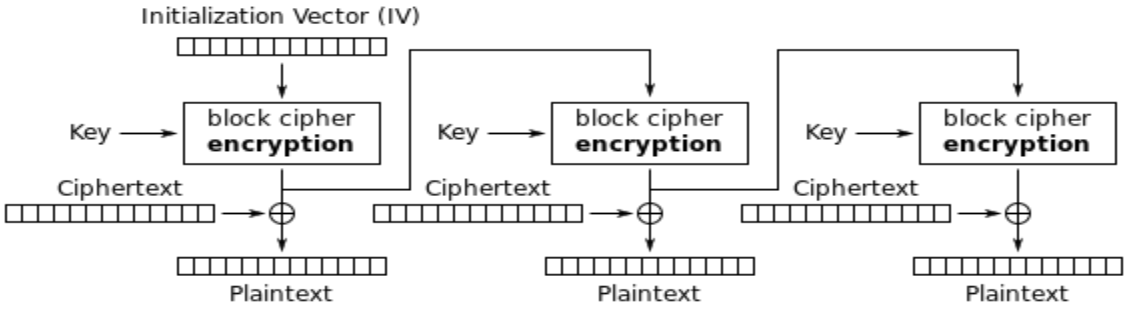
☐ Parallel encryption is not possible since every encryption requires a previous cipher.

c)Output Feedback Mode(OFB):–

The output feedback mode follows nearly the same process as the Cipher Feedback mode except that it sends the encrypted output as feedback instead of the actual cipher which is XOR output. In this output feedback mode, all bits of the block are sent instead of sending selected s bits. The Output Feedback mode of block cipher holds great resistance towards bit transmission errors. It also decreases the dependency or relationship of the cipher on the plaintext.



Output Feedback (OFB) mode encryption



Output Feedback (OFB) mode decryption

Advantages of OFB:–

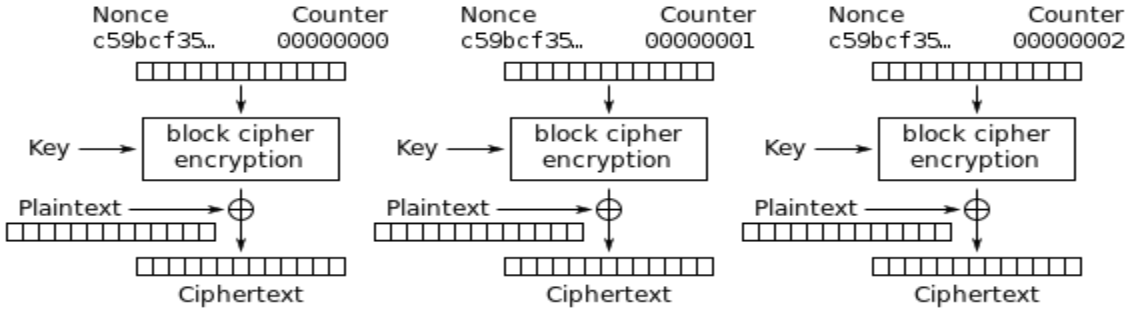
☐ In the case of CFB, a single bit error in a block is propagated to all subsequent blocks. This problem is solved by OFB as it is free from bit errors in the plaintext block.

Disadvantages of OFB:–

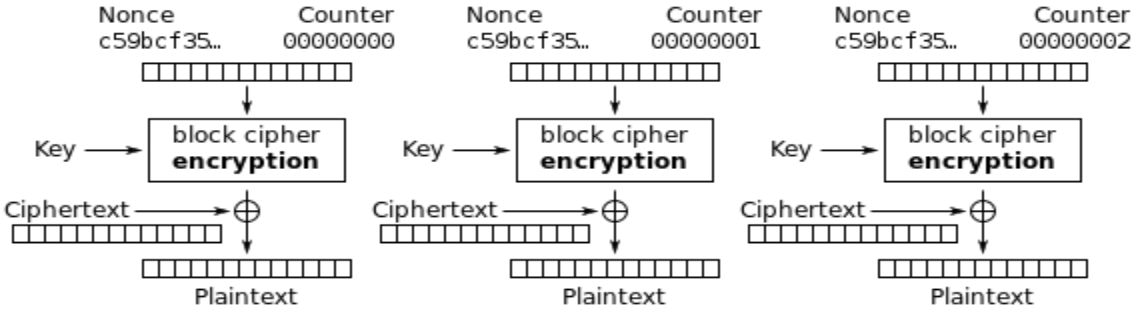
☐ The drawback of OFB is that, because to its operational modes, it is more susceptible to a message stream modiiccation attack than CFB. **d)Counter Mode:–**

The Counter Mode or CTR is a simple counter-based block cipher implementation. Every time a counter-initiated value is encrypted and given as input to XOR with plaintext which results in ciphertext block. The CTR mode is independent of feedback use and thus can be implemented in parallel.

Its simple implementation is shown below:



Counter (CTR) mode encryption



Counter (CTR) mode decryption

Advantages of Counter:–

- Since there is a different counter value for each block, the direct plaintext and ciphertext relationship is avoided. This means that the same plain text can map to different ciphertext.
- Parallel execution of encryption is possible as outputs from previous stages are not chained as in the case of CBC.

Disadvantages of Counter:-

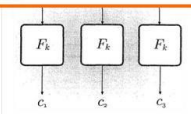
☐ The fact that CTR mode requires a synchronous counter at both the transmitter and the receiver is a severe drawback. The recovery of plaintext is erroneous when synchronisation is lost.

OUPUT FOR EVERY MODE:

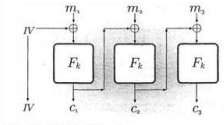
Activities Firefox Web Browser Tue 15:09

Virtual Labs | check answer in ecb in vi | https://cse29-iiith.vlabs.ac.in/exp/aes/theory.html 50%

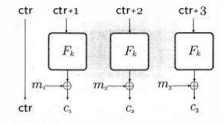
Assignment
References
Feedback



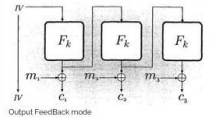
Electronic Code Book(ECB) mode



Cipher Block Chaining(CBC) mode



Contur mode



Output FeedBack mode

Community Links | Contact Us | Follow Us

Activities Firefox Web Browser Tue 15:08

Virtual Labs | check answer in ecb in vi | https://cse29-iiith.vlabs.ac.in/exp/aes/simulation.html 80%

AES and Modes of Operation

PART I

Choose your mode of operation: Electronic Code Book (ECB)

PART II

Key size in bits: 128

Plaintext: 789dd094 1ed35c5a 38d6d4c7 75697125
feea1f33 af3ba572 777db0d1 79139fd1
52930f25 70d74f6a 87e0105b b8f3cdf5
526f05e8 859547d0 faac25a9 d1c522f4
f8ba699b 569815ed 66a49d3d 7b53d172

Next Plaintext Key: 2f57afd1 2738c036 3ddf8695 85802789 Next Keytext

IV: Next IV

CTR: Next CTR

PART III

Calculate XOR:

Calculate XOR

XOR:

PART IV

Key in hex: 2f57afd1 2738c036 3ddf8695 85802789

Plaintext in hex: f8ba699b 569815ed 66a49d3d 7b53d172

Ciphertext in hex: 3e6c556b 9cd03380 48b0e055 79ab7a2c

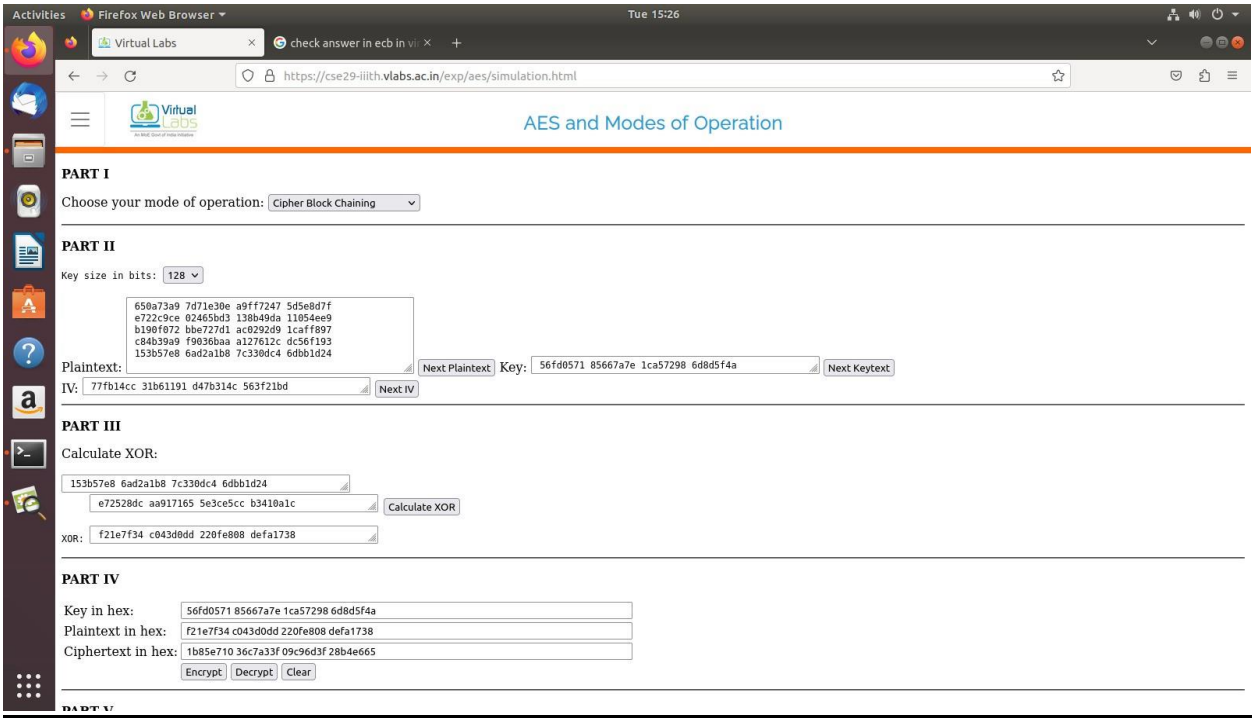
Encrypt Decrypt Clear

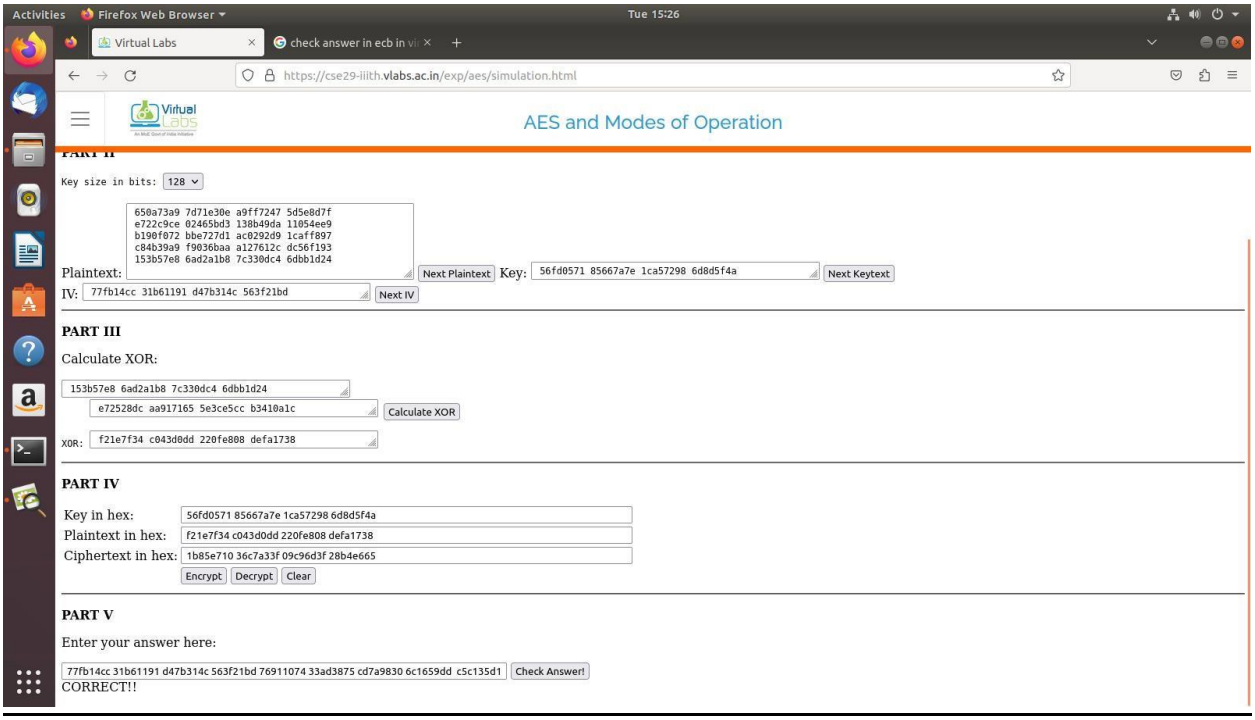
PART V

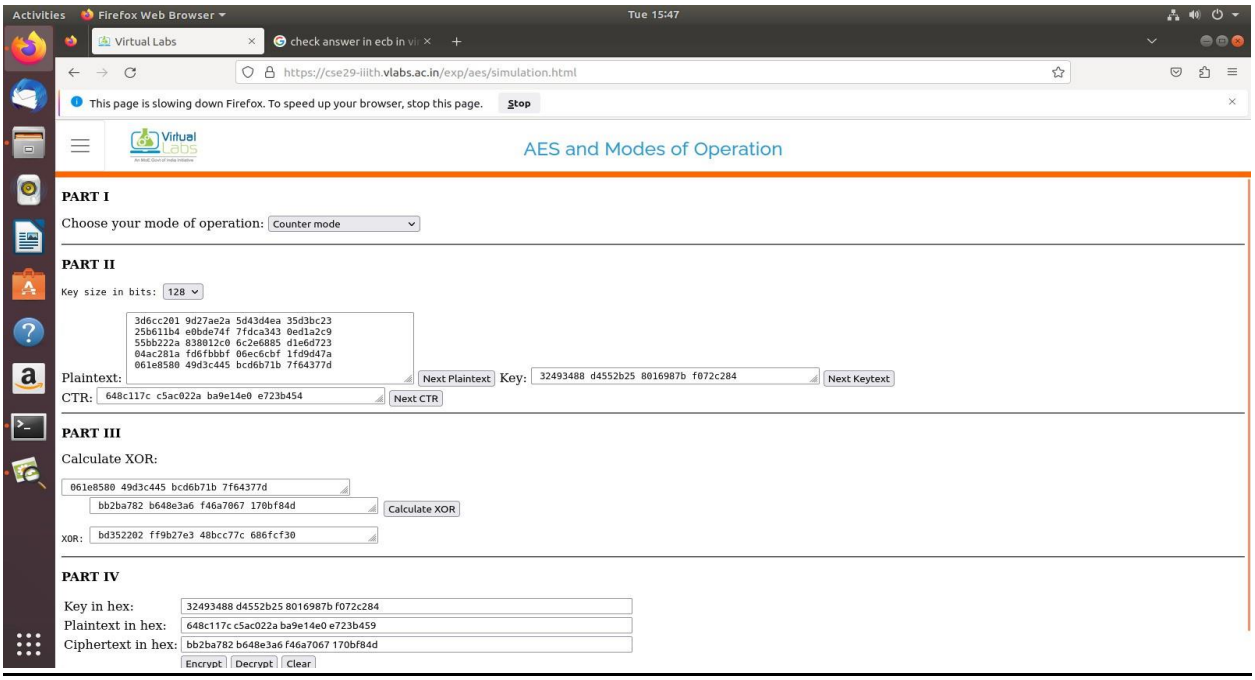
Enter your answer here:

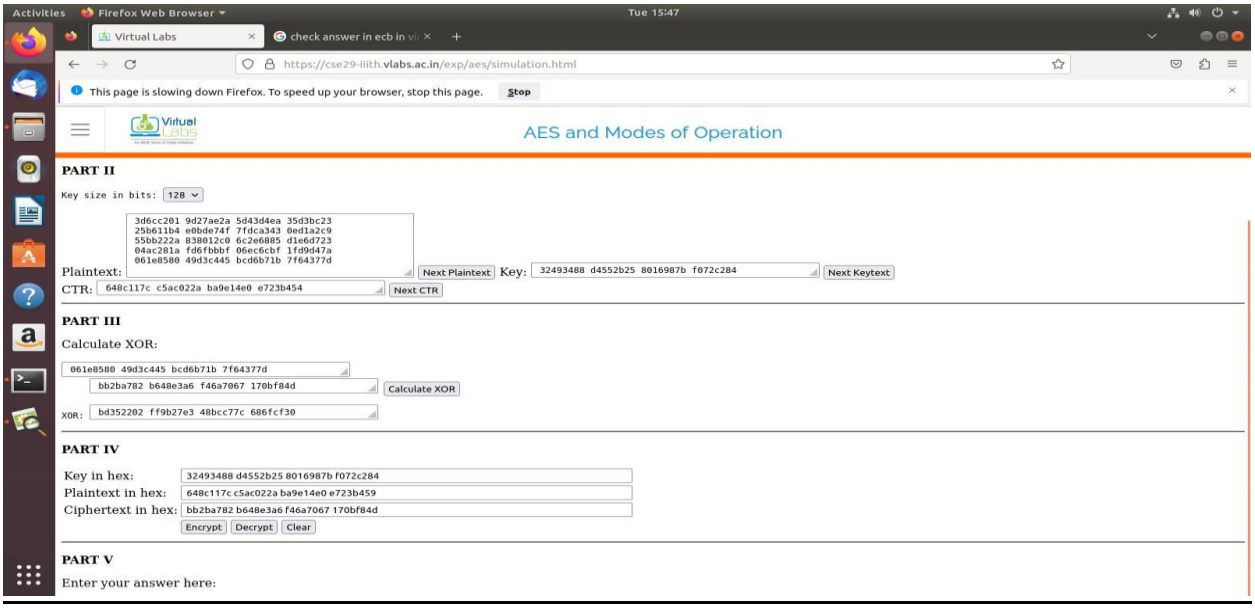
14bca2b 45bc5adc 12f7b066 6e93ffd5 fc43c89a a5893c99 3e6c556b 9cd03380 48b0e0 Check Answer!

CORRECT!!


Activities
Firefox Web Browser
Tue 15:26
Virtual Labs
check answer in ecb in vi
https://cse29-iiith.vlabs.ac.in/exp/aes/simulation.html
AES and Modes of Operation
PART I
Choose your mode of operation: Cipher Block Chaining
PART II
Key size in bits: 128
650a73a9 7d71e30e a9ff7247 5d5e8d7f
e722c9ce 02465bd3 138b49da 11054ee9
b190f072 bbe727d1 ac0292d9 1caff897
c84b39a9 f9036baa a127612c dc56f193
153b57e8 6ad2a1b8 7c330dc4 6dbb1d24
Plaintext:
Next Plaintext
Key: 56fd0571 85667a7e 1ca57298 6d8d5f4a
Next Keytext
IV: 77fb14cc 31b61191 d47b314c 563f21bd
Next IV
PART III
Calculate XOR:
153b57e8 6ad2a1b8 7c330dc4 6dbb1d24
e72528dc aa917165 5e3ce5cc b3410a1c
Calculate XOR
XOR: f21e7f34 c043d0dd 220fe808 defa1738
PART IV
Key in hex: 56fd0571 85667a7e 1ca57298 6d8d5f4a
Plaintext in hex: f21e7f34 c043d0dd 220fe808 defa1738
Ciphertext in hex: 1b85e710 36c7a33f 09c96d3f 28b4e665
Encrypt
Decrypt
Clear


Activities
Firefox Web Browser
Tue 15:26
Virtual Labs
check answer in ecb in vi
https://cse29-iiith.vlabs.ac.in/exp/aes/simulation.html
AES and Modes of Operation
Key size in bits: 128
650a73a9 7d71e30e a9ff7247 5d5e8d7f
e722c9ce 02465bd3 138b49da 11054ee9
b190f072 bbe727d1 ac0292d9 1caff897
c84b39a9 f9036baa a127612c dc56f193
153b57e8 6ad2a1b8 7c330dc4 6dbb1d24
Plaintext:
Next Plaintext
Key: 56fd0571 85667a7e 1ca57298 6d8d5f4a
Next Keytext
IV: 77fb14cc 31b61191 d47b314c 563f21bd
Next IV
PART III
Calculate XOR:
153b57e8 6ad2a1b8 7c330dc4 6dbb1d24
e72528dc aa917165 5e3ce5cc b3410a1c
Calculate XOR
XOR: f21e7f34 c043d0dd 220fe808 defa1738
PART IV
Key in hex: 56fd0571 85667a7e 1ca57298 6d8d5f4a
Plaintext in hex: f21e7f34 c043d0dd 220fe808 defa1738
Ciphertext in hex: 1b85e710 36c7a33f 09c96d3f 28b4e665
Encrypt
Decrypt
Clear
PART V
Enter your answer here:
77fb14cc 31b61191 d47b314c 563f21bd 76911074 33ad3875 cd7a9830 6c1659dd c5c135d1
Check Answer!
CORRECT!!





CONCLUSION: Successfully, we learnt utilizing the AES cipher with various block cipher modes, it is crucial to carefully evaluate the speciicc security and operational requirements. Always implement the chosen mode correctly and adhere to best practices to uphold the security of your data.

Assignment 3

Aim:- Implementation and analysis of RSA cryptosystem and Digital Signature scheme using RSA

Lo Mapped:- Lo2

Theory :- RSA (Rivest-Shamir-Adleman):-

RSA is a widely used public-key cryptosystem for secure data transmission and digital signatures. It's based on the mathematical properties of large prime numbers. RSA involves a pair of keys: a public key for encryption and a private key for decryption. The security of RSA relies on the difficulty of factoring large semiprime numbers.

Algorithm:-

1. Key Generation:

   - Choose two distinct prime numbers, p and q.

   - Calculate n = p * q.

   - Compute the totient φ(n) = (p - 1) * (q - 1).

   - Choose an integer e (usually a small prime, commonly 65537) that is coprime with φ(n).

   - Compute d such that (d * e) % φ(n) = 1.

   - Public key: (e, n)

   - Private key: (d, n)

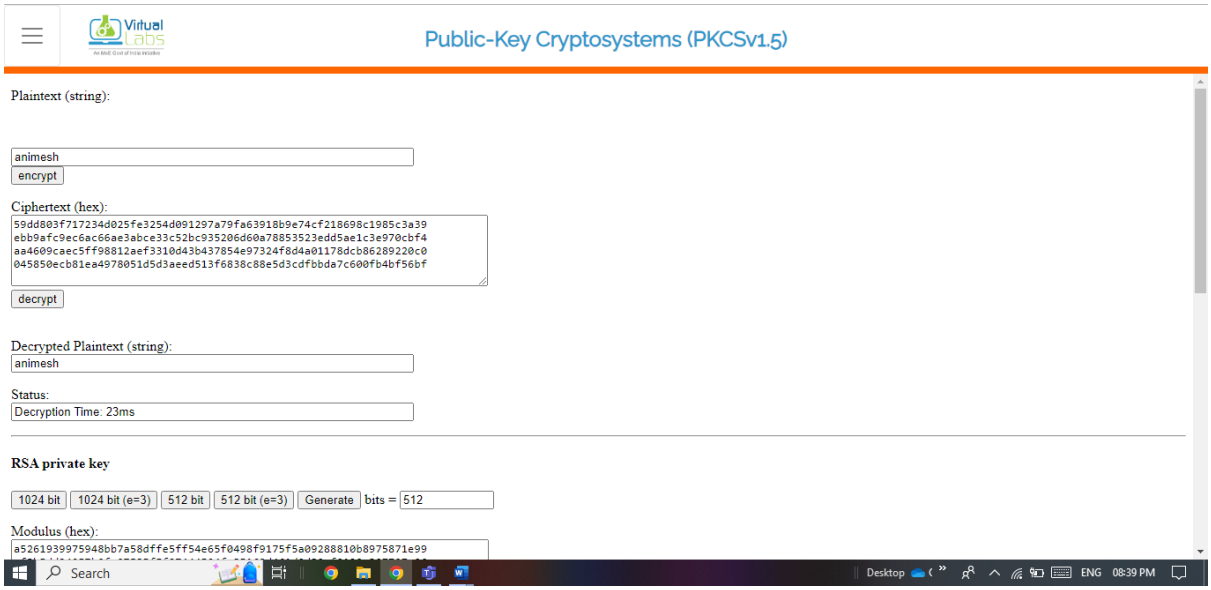
2. Encryption:

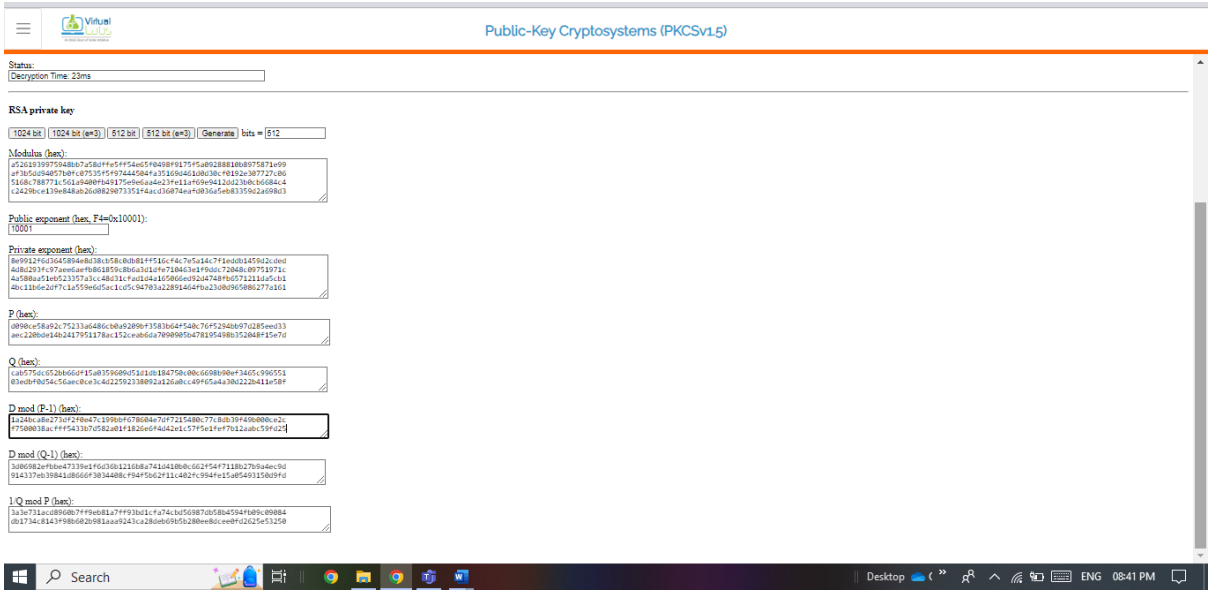
   - Convert the plaintext message into a numeric value m.

   - Compute the ciphertext c = (m^e) % n.

3. Decryption:

   - Compute the plaintext message m = (c^d) % n.





Digital Signature:-

A digital signature is a cryptographic technique that provides authenticity, integrity, and non-repudiation for digital messages or documents. It involves using a private key to sign the message and a public key to verify the signature. Digital signatures ensure that the sender of a message is authenticated and that the message has not been tampered with during transmission.

Algorithm:-

1. Key Generation:

   - Choose a private key for signing.

   - Compute a corresponding public key for verification.

2. Signing:

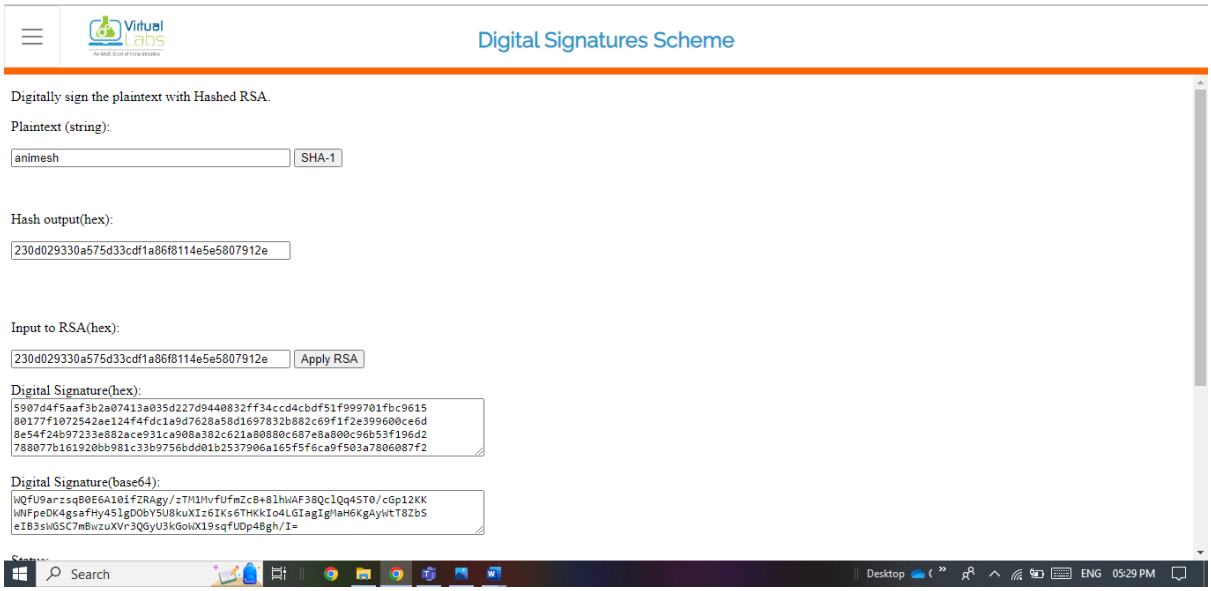
   - Hash the message to produce a fixed-length digest.

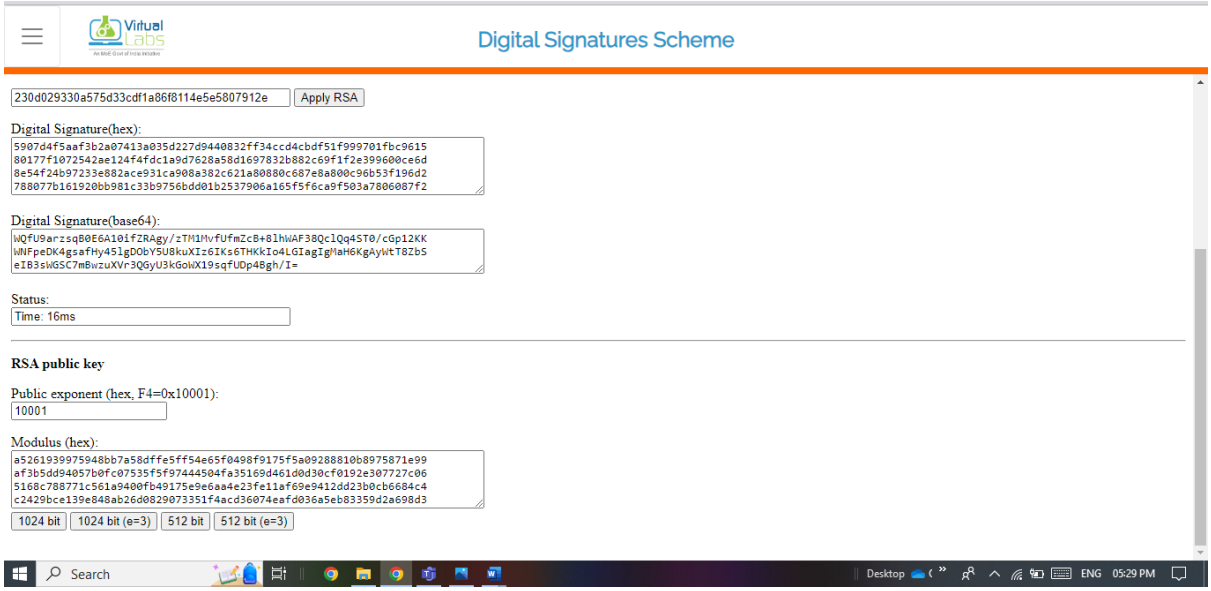
- Encrypt the digest using the private key to create the digital signature.

3. Verification:

- Decrypt the digital signature using the sender's public key to get the digest.
- Hash the received message to produce a digest.
- Compare the two digests. If they match, the signature is valid.

Digital signatures are essential for secure communication, online transactions, and authentication of digital documents.





Conclusion:- Thus we learnt and implemented RSA and digital signature using RSA

Animesh Parab t2-t21 88

Lab Assignment 5

Aim: To explore Hashdeep tool in kali linux for generating, matching and auditing hash of files.

Lab Outcome Attainment: LO2

Theory:

Hashdeep is a command-line utility for computing and verifying hash values (checksums) of files and directories. It is a versatile and powerful tool primarily used for data integrity verification and digital forensics. Hashdeep can calculate multiple hash values (e.g., MD5, SHA-1, SHA-256, SHA-512) for files and directories and store them in hash databases. You can then use these hash databases to verify the integrity of your files at a later time by comparing the computed hash values with the stored ones. Some key features and use cases of Hashdeep include:

1. **Data Integrity Verification**: Hashdeep is commonly used to ensure that files have not been tampered with or corrupted over time. By periodically recalculating hash values and comparing them to the stored values, you can detect any unauthorized changes.

2. **Forensics and Investigations**: Digital forensics experts use Hashdeep to create hash databases of evidence and verify its integrity during investigations. This helps ensure that the data remains unchanged throughout the legal process.

3. **Comparing Directories**: You can use Hashdeep to compare two directories to find differences between them, even if the file names have changed. This is useful for backup verification and synchronization tasks.

4. **Recursive Hashing**: Hashdeep can recursively calculate hash values for directories and subdirectories, making it efficient for processing large and complex directory structures.

5. **Cross-Platform**: Hashdeep is available for various operating systems, including Linux, macOS, and Windows, making it a versatile tool for cross-platform use.
6. **Support for Multiple Hash Algorithms**: It supports multiple hash algorithms, including MD5, SHA-1, SHA-256, SHA-512, and others, allowing you to choose the level of security and performance you need.

How to use **hashdeep :**

1. To check the version of Hashdeep - *Hashdeep -V*

2. To display help about Hashdeep - *Hashdeep -h* or *Hashdeep -hh*

3. To display the manual page of Hashdeep- *man Hashdeep*

4. To display the manual page of any specific hash algorithm supported by Hashdeep- *man md5deep*

5. To hash a file - *Hashdeep filename*

6. To supress any error messages- *Hashdeep -s filename*

7. To apply multiple hash algorithms than default-

   Hashdeep -c md5,sha1,sha256,tiger filename

8. To hash multiple files (say all text files) using md5

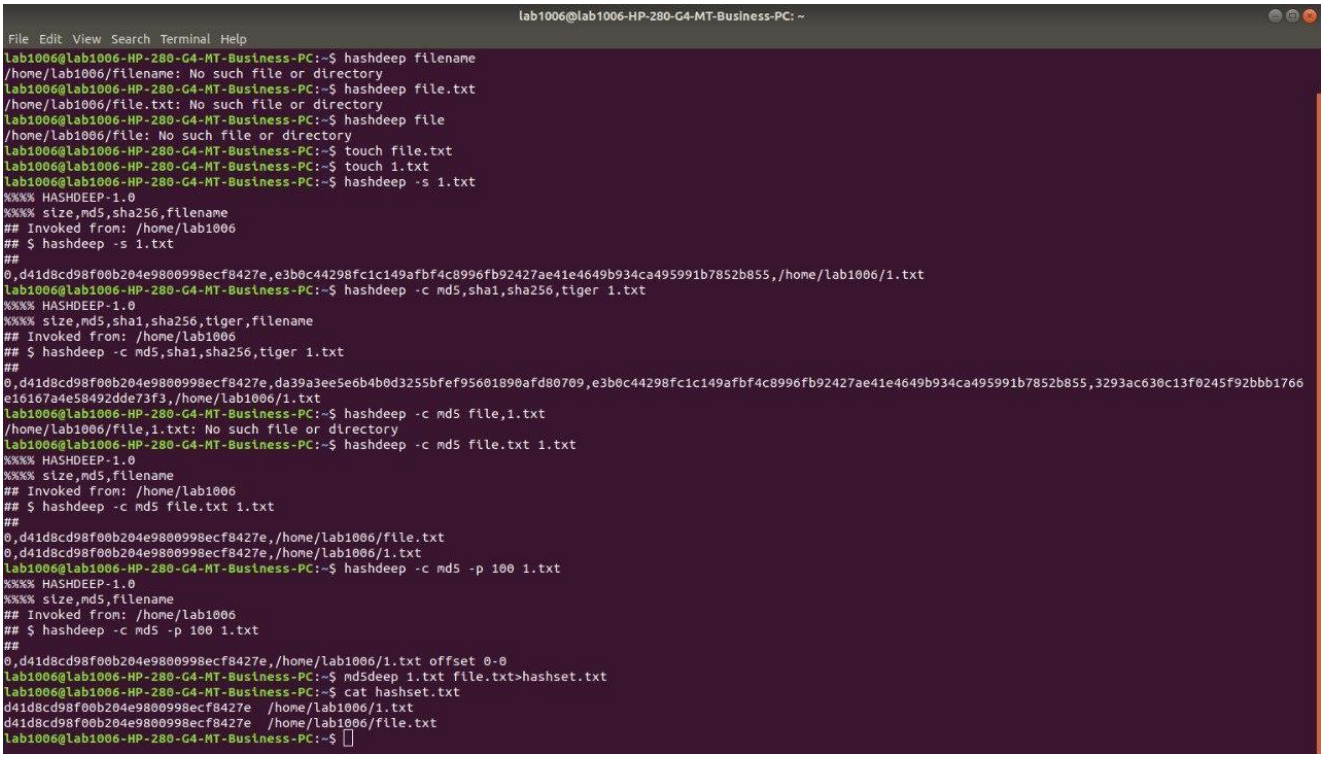
   *Hashdeep -c md5 *.txt*

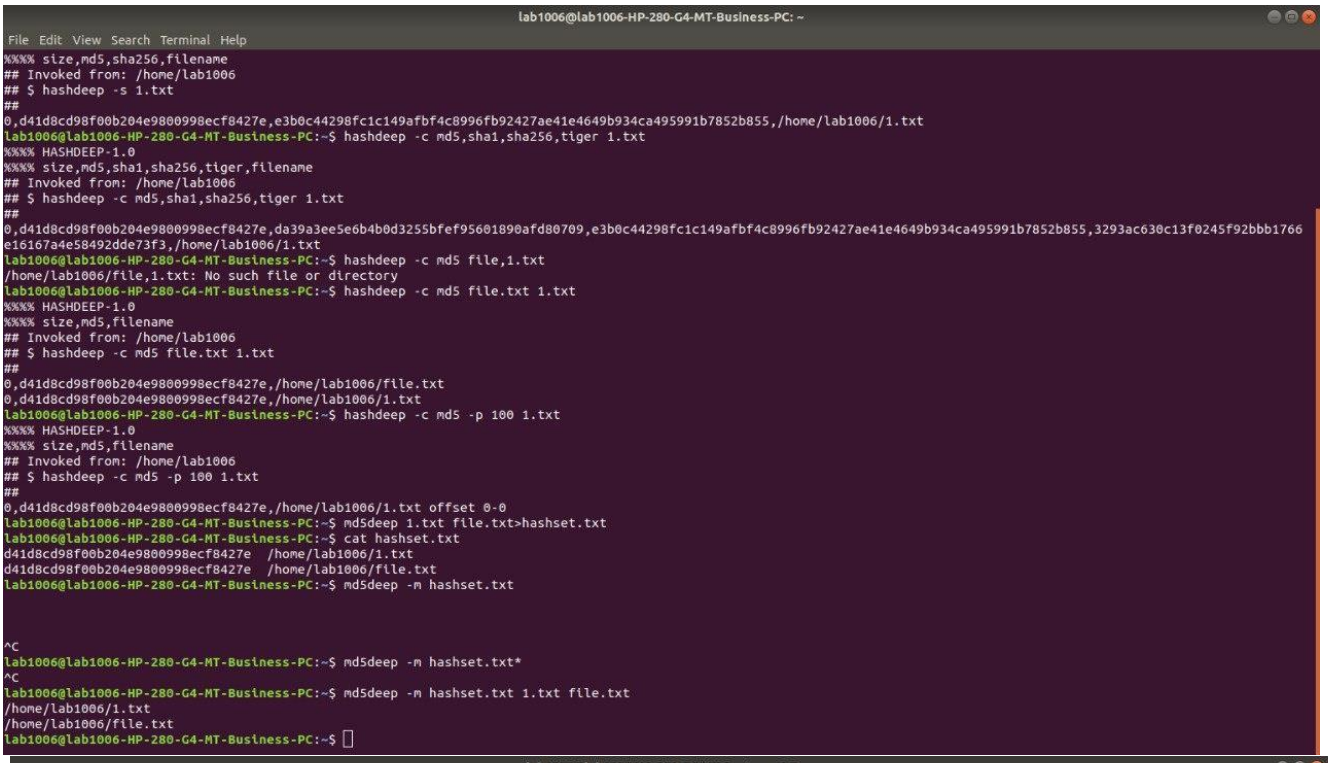
9. To hash multiple files (say all text files) using md5 and sha1

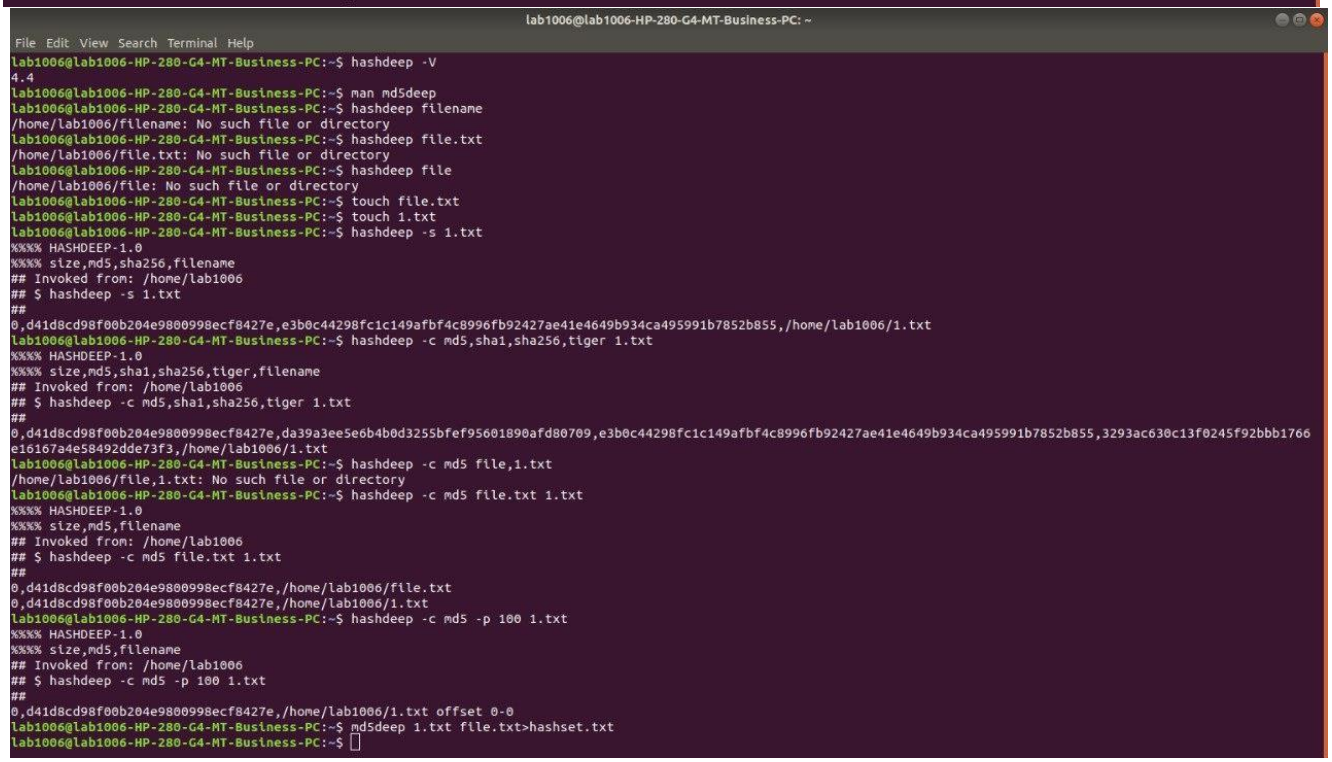
   *Hashdeep -c md5,sha1 *.txt*

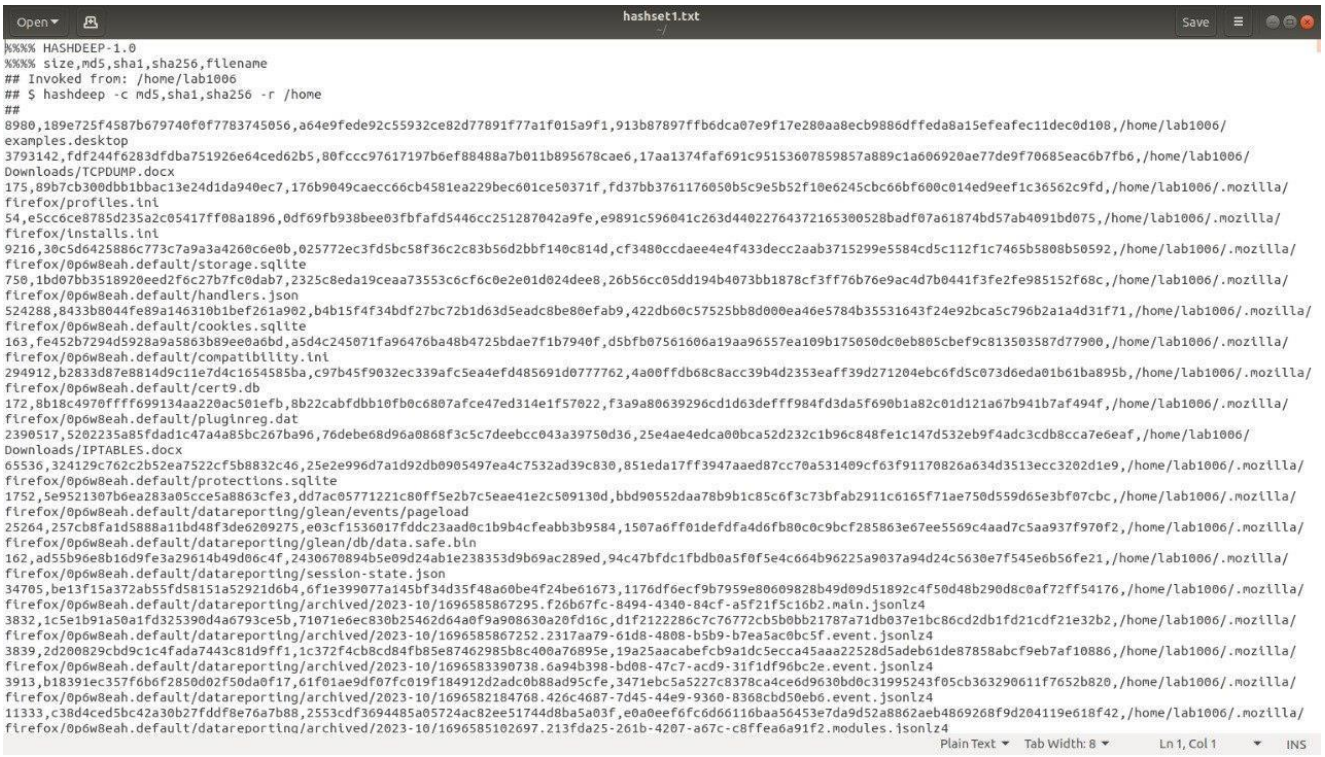
10. Hashing block of files-
    Hashdeep -c md5 -p 100 example.txt

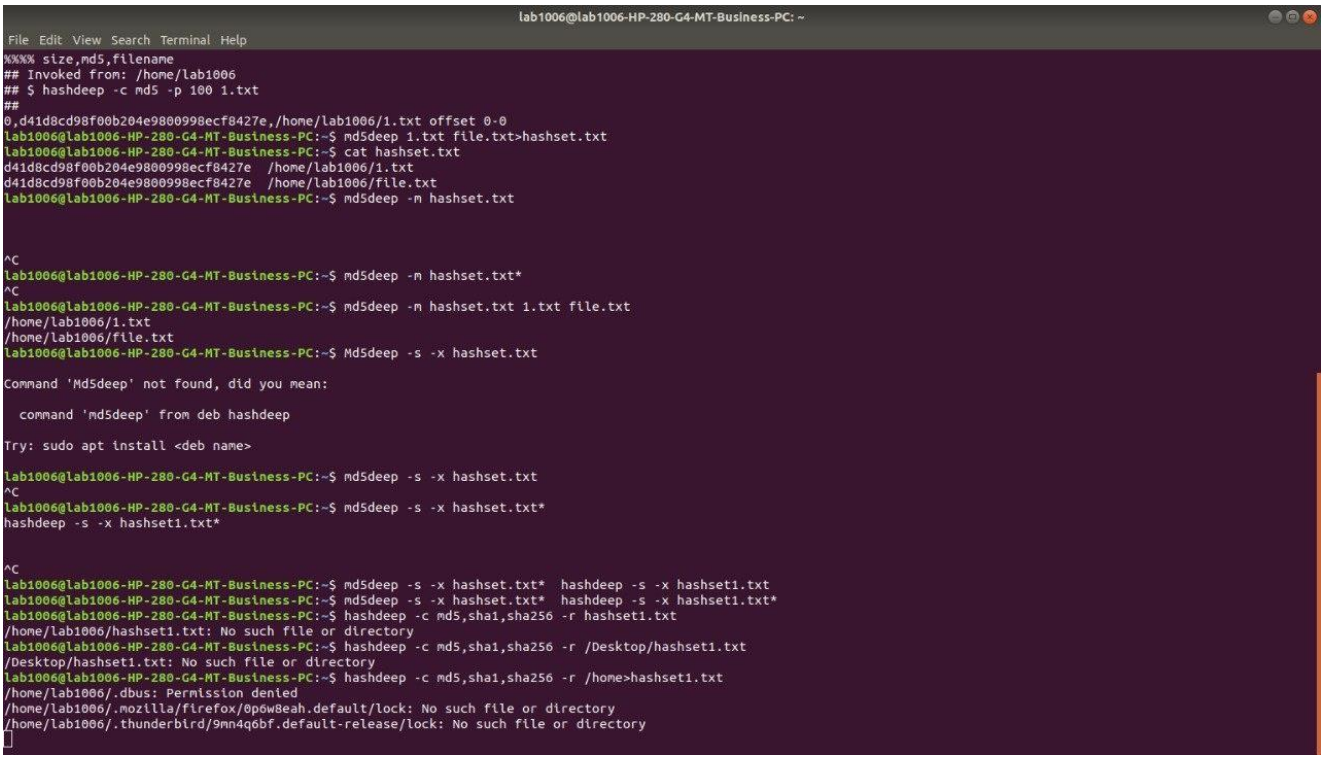
Output :




lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
%%%% size,md5,sha256,filename
Invoked from: /home/lab1006
$ hashdeep -s 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5,sha1,sha256,tiger 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha1,sha256,tiger,filename
Invoked from: /home/lab1006
$ hashdeep -c md5,sha1,sha256,tiger 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,da39a3ee5e6b4b0d3255bfef95601890afd80709,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,3293ac630c13f0245f92bbb1766
e16167a4e58492dde73f3,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 file,1.txt
/home/lab1006/file,1.txt: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 file.txt 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
Invoked from: /home/lab1006
$ hashdeep -c md5 file.txt 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/file.txt
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 -p 100 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
Invoked from: /home/lab1006
$ hashdeep -c md5 -p 100 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/1.txt offset 0-0
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ md5deep 1.txt file.txt>hashset.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ cat hashset.txt
d41d8cd98f00b204e9800998ecf8427e /home/lab1006/1.txt
d41d8cd98f00b204e9800998ecf8427e /home/lab1006/file.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ md5deep -m hashset.txt
^C
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ md5deep -m hashset.txt*
^C
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ md5deep -m hashset.txt 1.txt file.txt
/home/lab1006/1.txt
/home/lab1006/file.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$


lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -V
4.4
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man md5deep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep filename
/home/lab1006/filename: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep file.txt
/home/lab1006/file.txt: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep file
/home/lab1006/file: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ touch file.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ touch 1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -s 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
Invoked from: /home/lab1006
$ hashdeep -s 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5,sha1,sha256,tiger 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha1,sha256,tiger,filename
Invoked from: /home/lab1006
$ hashdeep -c md5,sha1,sha256,tiger 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,da39a3ee5e6b4b0d3255bfef95601890afd80709,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,3293ac630c13f0245f92bbb1766
e16167a4e58492dde73f3,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 file,1.txt
/home/lab1006/file,1.txt: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 file.txt 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
Invoked from: /home/lab1006
$ hashdeep -c md5 file.txt 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/file.txt
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 -p 100 1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
Invoked from: /home/lab1006
$ hashdeep -c md5 -p 100 1.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,/home/lab1006/1.txt offset 0-0
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ md5deep 1.txt file.txt>hashset.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$





Conclusion :

We understood hashdeep and its versatile command-line utility that computes and verifies checksums (hash values) for files and directories, offering data integrity assurance and digital forensics capabilities. It supports multiple hash algorithms, making it a reliable tool for detecting file tampering and ensuring the integrity of data across different platforms.

Assignment 4

Aim: Study the use of network reconnaissance tools like WHOIS, dig, traceroute, nslookup, nikto, dmitry, to gather information about networks and domain registrars.

LO Mapped: LO3

Theory:

- WHOIS

The whois command displays information about a website's record. You may get all the information about a website regarding its registration and owner's information.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ whois google.com
   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.markmonitor.com
   Registrar URL: http://www.markmonitor.com
   Updated Date: 2019-09-09T15:39:04Z
   Creation Date: 1997-09-15T04:00:00Z
   Registry Expiry Date: 2028-09-14T04:00:00Z
   Registrar: MarkMonitor Inc.
   Registrar IANA ID: 292
   Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
   Registrar Abuse Contact Phone: +1.2086851750
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2023-08-08T05:48:12Z <<<
```

- dig

dig command stands for **Domain Information Groper**. It is used for retrieving information about DNS name servers. It is basically used by network administrators. It is used for verifying and troubleshooting DNS problems and to perform DNS lookups.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ dig www.google.com

; <<>> DiG 9.11.3-1ubuntu1.18-Ubuntu <<>> www.google.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27441
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;www.google.com.                        IN      A

;; ANSWER SECTION:
www.google.com.         75      IN      A       142.250.192.132

;; Query time: 3 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Tue Aug 08 11:29:33 IST 2023
;; MSG SIZE  rcvd: 59
```

- Traceroute

traceroute command in Linux prints the route that a packet takes to reach the host. This command is useful when you want to know about the route and about all the hops that a packet takes.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ traceroute google.com
traceroute to google.com (142.251.42.14), 30 hops max, 60 byte packets
 1  _gateway (192.168.0.1)  0.713 ms  0.711 ms  0.694 ms
 2  203.212.25.1 (203.212.25.1)  2.584 ms  2.592 ms  2.813 ms
 3  203.212.24.53 (203.212.24.53)  2.560 ms  2.552 ms  3.053 ms
 4  * * 10.10.226.153 (10.10.226.153)  4.219 ms
 5  72.14.196.213 (72.14.196.213)  8.154 ms  8.143 ms  4.715 ms
 6  108.170.248.177 (108.170.248.177)  4.904 ms  5.474 ms  5.204 ms
 7  209.85.250.139 (209.85.250.139)  4.781 ms  2.042 ms  2.030 ms
 8  bom12s19-in-f14.1e100.net (142.251.42.14)  2.126 ms  2.289 ms  2.755 ms
```

- nslookup

The nslookup command **queries internet domain name servers in two modes**. Interactive mode allows you to query name servers for information about various hosts and domains, or to print a list of the hosts in a domain.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ nslookup google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 142.251.42.14
Name:   google.com
Address: 2404:6800:4009:82f::200e
```

- nikto

Nikto is an open source web server and web application scanner. Nikto can perform comprehensive tests against web servers for multiple security threats, including over 6700 potentially dangerous files/programs. Nikto can also perform checks for outdated web servers software, and version-specific problems.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ nikto -h facebook.com
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          157.240.192.35
+ Target Hostname:    facebook.com
+ Target Port:        80
+ Start Time:         2023-08-08 11:54:26 (GMT5.5)
---------------------------------------------------------------------------
+ Server: proxygen-bolt
+ The anti-clickjacking X-Frame-Options header is not present.
+ Root page / redirects to: https://facebook.com/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Uncommon header 'proxy-status' found, with contents: http_request_error; e_clientaddr="AcLHSJ5kd9scY9Ol-zV7mMz9J6eFGG4gXftcTKNsDdkDmXSUcRbJ2nGiv_7tHwQnA4jVNpHWWbGPy10
0sdw3"; e_fb_vipaddr="AcJwsHMwiQCLSif1b7SUxciHeBFfdSYSW1L2ZH7vu_pVS5UIWbAxlLRu2MgV_5W_UZuOiHpaNRo"; e_fb_builduser="AcI_kbko1ZVnsYpCBYc17BqMhEI_iSAJnbhpR21CLdqX_6-4q-3j
LCG90z_pY9h5SDE"; e_fb_binaryversion="AcJ_73IhdWgJ_X0eWF9RCd5OVod35-dSK1tG2RZIM4ZISC4BdSRO0YUkiOd90SlGlaqnI02f24A8Q_JabP5x1R9bxog0y1Yzouc"; e_proxy="AcIFqBFJpc1Eb5F2iXd
TkwMxfB3Y6HYd5ZZYnGhjfTAS9bGIT_b0fZc2aeZ2HNr5MwsudQwNHh-rK5E"
+ 6544 items checked: 0 error(s) and 2 item(s) reported on remote host
+ End Time:           2023-08-08 11:57:25 (GMT5.5) (179 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

- dmitry

dmitry can find possible subdomains, email addresses, uptime information, perform tcp port scan, whois lookups, and more.

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ dmitry google.com
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:142.251.42.14
HostName:google.com

Gathered Inet-whois information for 142.251.42.14
---------------------------------

inetnum:        142.248.0.0 - 143.46.255.255
netname:        NON-RIPE-NCC-MANAGED-ADDRESS-BLOCK
descr:          IPv4 address block not managed by the RIPE NCC
remarks:        -----------------------------------------------------
remarks:
remarks:        For registration information,
remarks:        you can consult the following sources:
remarks:
remarks:        IANA
remarks:        http://www.iana.org/assignments/ipv4-address-space
remarks:        http://www.iana.org/assignments/iana-ipv4-special-registry
remarks:        http://www.iana.org/assignments/ipv4-recovered-address-space
remarks:
remarks:        AFRINIC (Africa)
remarks:        http://www.afrinic.net/ whois.afrinic.net
remarks:
remarks:        APNIC (Asia Pacific)
remarks:        http://www.apnic.net/ whois.apnic.net
remarks:
remarks:        ARIN (Northern America)
remarks:        http://www.arin.net/ whois.arin.net
remarks:
remarks:        LACNIC (Latin America and the Carribean)
remarks:        http://www.lacnic.net/ whois.lacnic.net
remarks:
remarks:        -----------------------------------------------------
country:        EU # Country is really world wide
admin-c:        IANA1-RIPE
tech-c:         IANA1-RIPE
status:         ALLOCATED UNSPECIFIED
mnt-by:         RIPE-NCC-HM-MNT
created:        2023-07-24T14:32:43Z
last-modified:  2023-07-24T14:32:43Z
source:         RIPE
```

Conclusion: In this experiment we used different network reconnaissance tools to gather information about the network.

Animesh Parab T2-T21 88

Lab Assignment 7

AIM: Study of packet sniffer tools TCPDUMP.

LO3: Explore the different network reconnaissance tools to gather information about networks.

THEORY:

What is TCPDUMP and how to install it?

Tcpdump is a command-line packet analyzer that allows you to capture and analyze network traffic in real-time. It's commonly used for troubleshooting network issues, analyzing network behavior, and diagnosing problems related to network communication. tcpdump captures packets as they travel through a network interface and provides detailed information about each packet, including source and destination addresses, protocol information, payload data, and more.

Linux (Debian/Ubuntu):

Open a terminal and run the following command to install tcpdump: sudo apt-get update sudo apt-get install tcpdump

Explain various commands in tcpdump to capture different types of packets.

tcpdump provides a wide range of commands and options to capture and analyze different types of packets. Here are some common tcpdump commands and filters to capture specific types of packets:

1. Capture All Traffic on a Specific Interface:

sudo tcpdump -i eth0

This captures all traffic on the "eth0" network interface.

2. Capture Traffic to or from a Specific IP Address:

sudo tcpdump host 192.168.1.100

This captures all traffic to or from the IP address "192.168.1.100".

3. Capture Traffic on a Specific Port:

sudo tcpdump port 80

This captures all traffic on port 80.

4. Capture Traffic Using a Specific Protocol:

sudo tcpdump icmp

This captures ICMP (ping) traffic.

5. Capture Traffic from a Specific Source IP:

sudo tcpdump src 192.168.1.200

This captures traffic originating from IP address "192.168.1.200".

6. Capture Traffic to a Specific Destination IP: sudo tcpdump dst 192.168.1.100

This captures traffic directed to IP address "192.168.1.100".

7. Capture Traffic on a Specific Port Using a Protocol:

sudo tcpdump udp port 53

This captures UDP traffic on port 53 (DNS).

8. Capture Traffic Using a Combination of Filters:

sudo tcpdump src 192.168.1.100 and port 22

This captures traffic originating from IP address "192.168.1.100" and using port 22 (SSH).

9. Capture Traffic with Specific Packet Size:

sudo tcpdump greater 1000

This captures packets larger than 1000 bytes.

10. Capture Specific Number of Packets:

sudo tcpdump -c 10

This captures 10 packets and then exits.

11. Capture Packets Using Hexadecimal Filter:

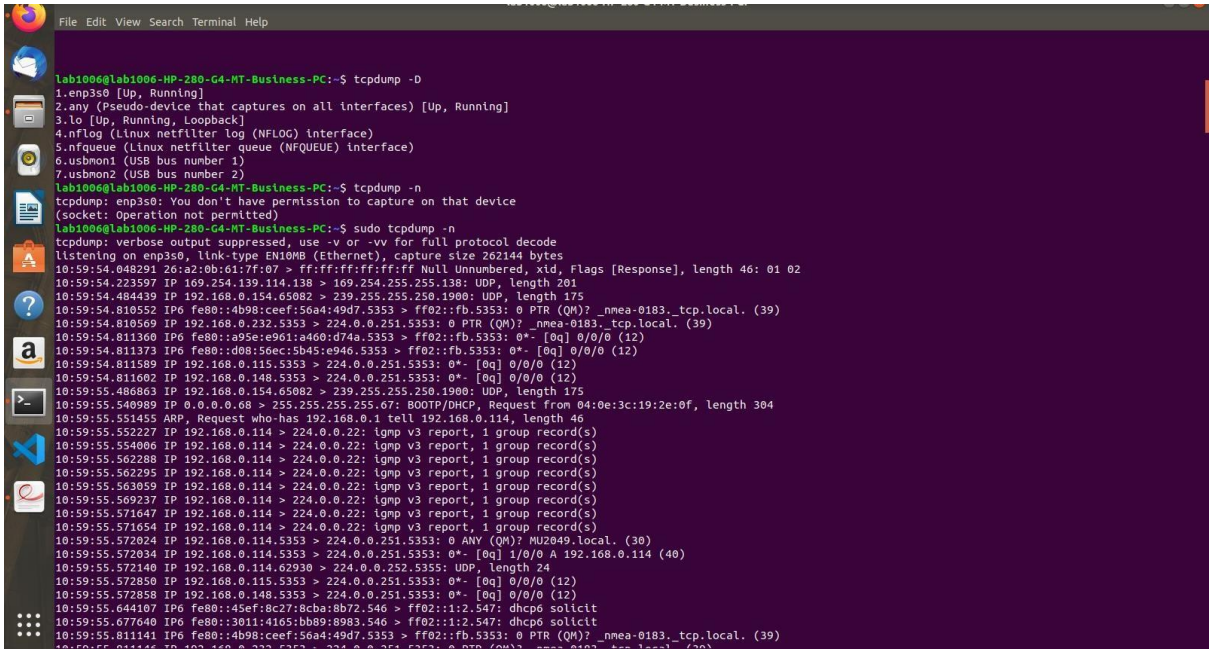
sudo tcpdump -X 'tcp[13] & 2 != 0'

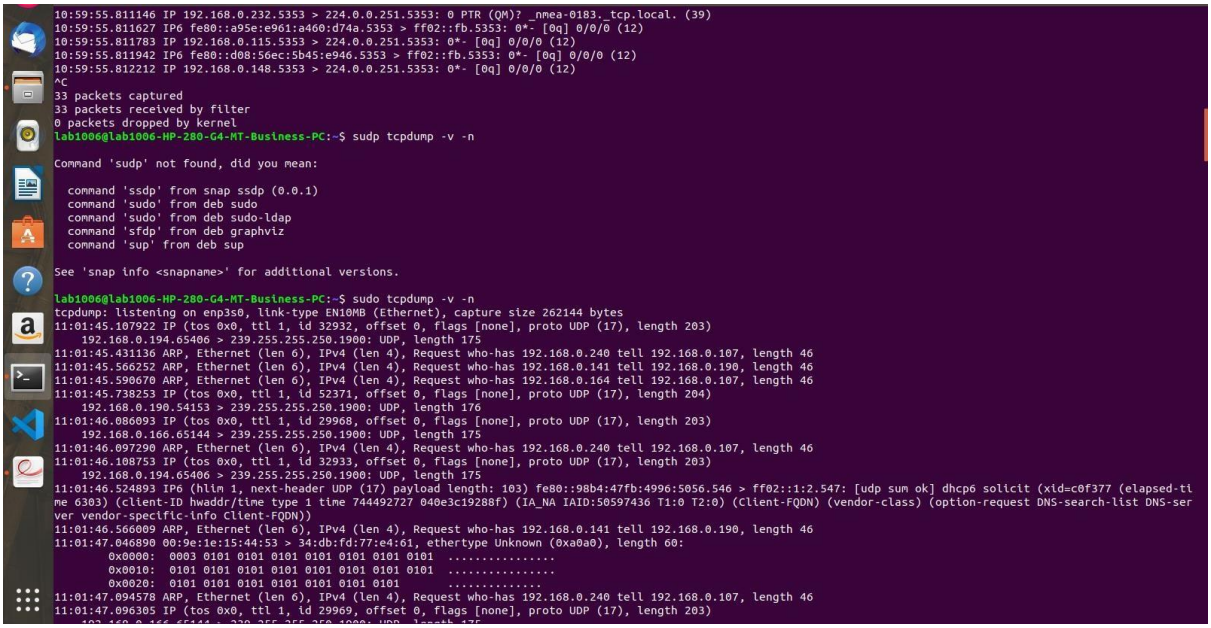
This captures only SYN packets (TCP packets with the SYN flag set).

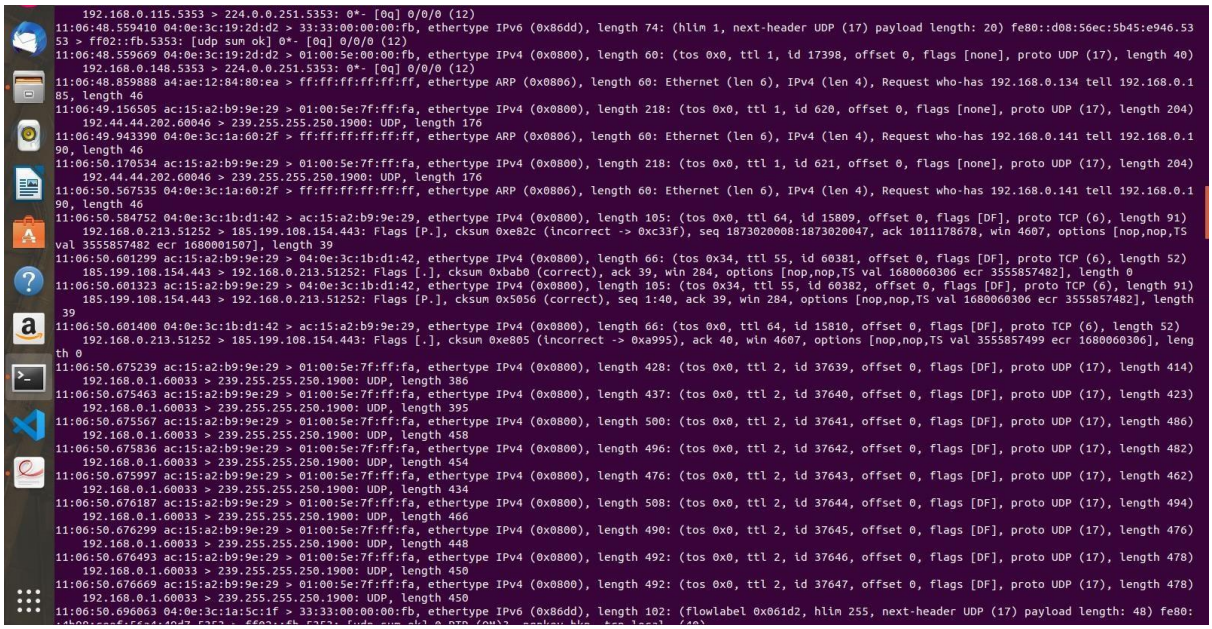
12. Capture and Save Output to a File: sudo tcpdump -i eth0 -w output.pcap

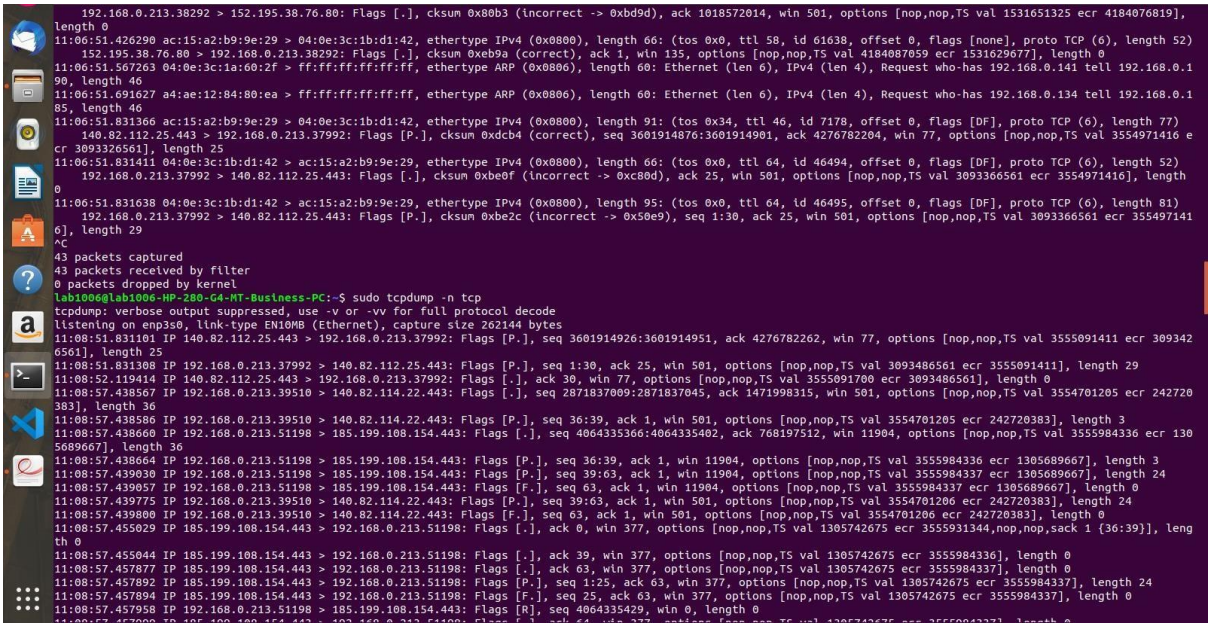
This captures traffic on the "eth0" interface and saves it to the "output.pcap" file.

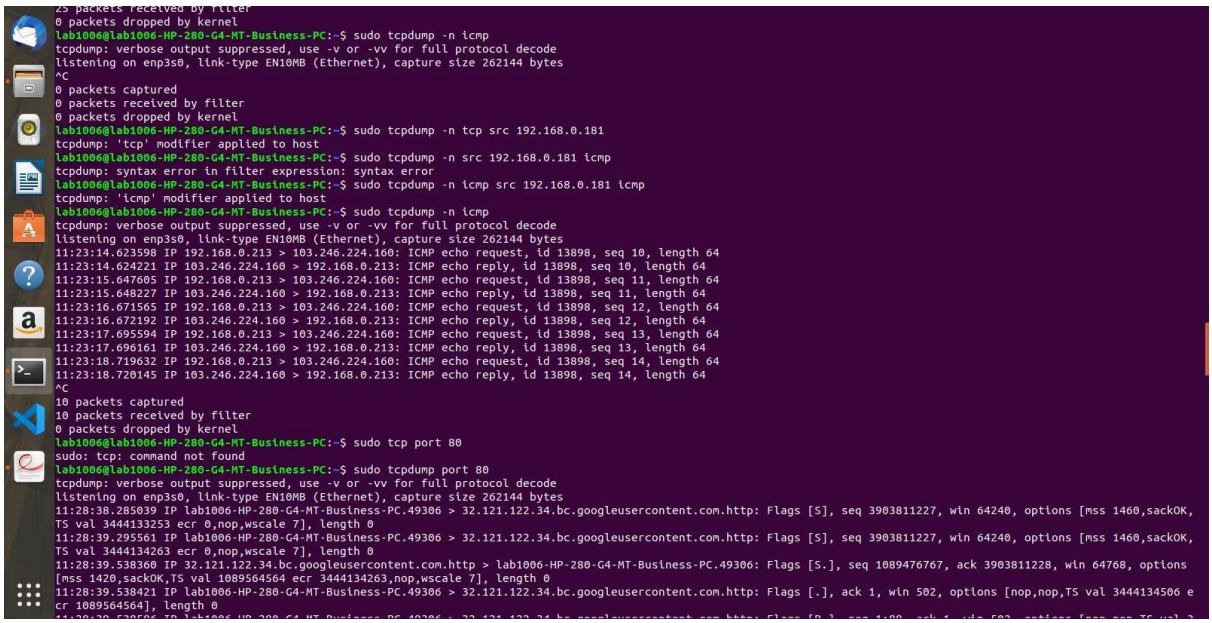
OUTPUT

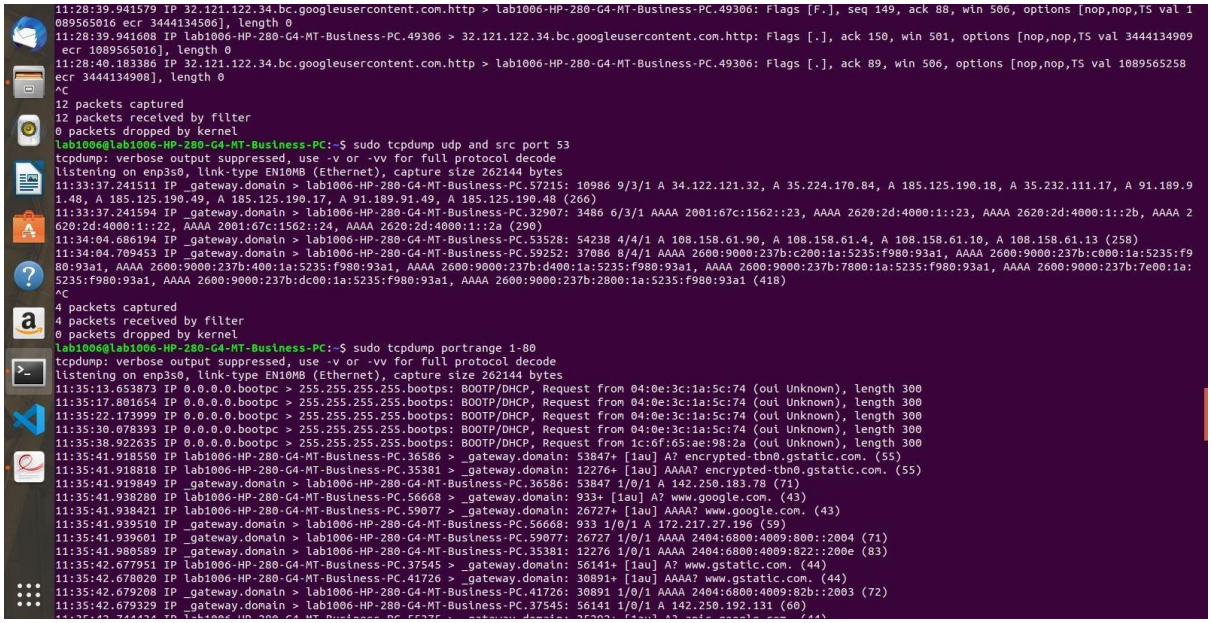


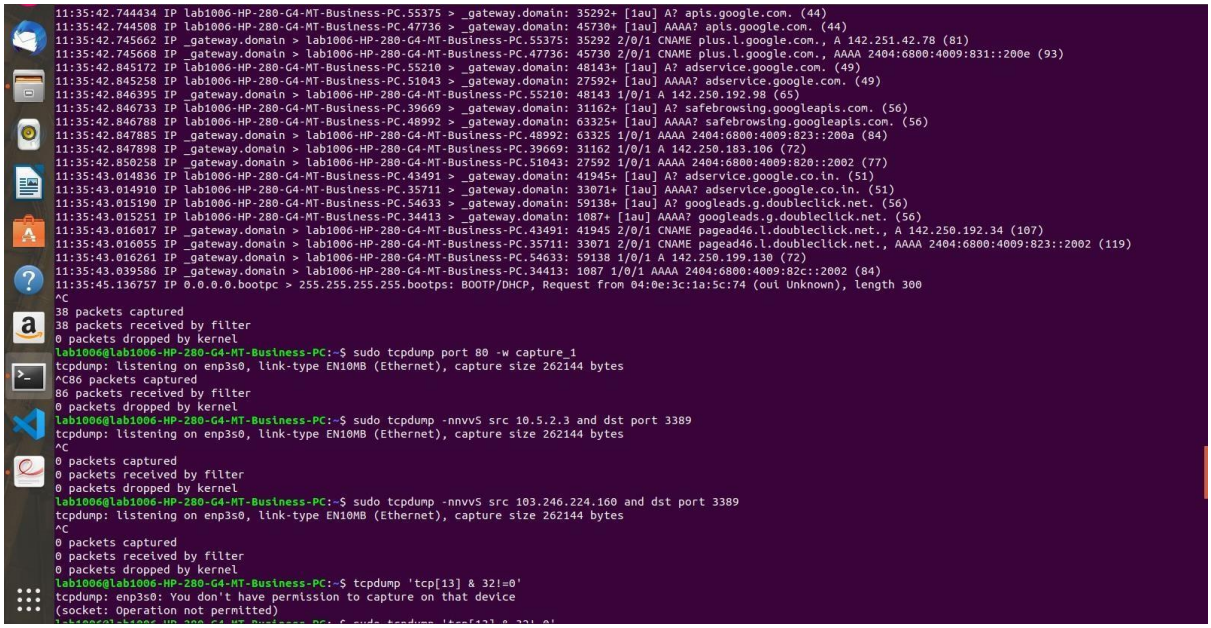

10:59:55.811146 IP 192.168.0.232.5353 > 224.0.0.251.5353: 0 PTR (QM)? _nmea-0183._tcp.local. (39)
10:59:55.811627 IP6 fe80::a95e:e961:a460:d74a.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
10:59:55.811783 IP 192.168.0.115.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
10:59:55.811942 IP6 fe80::d08:56ec:5b45:e946.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
10:59:55.812212 IP 192.168.0.148.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
^C
33 packets captured
33 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudp tcpdump -v -n
Command 'sudp' not found, did you mean:
command 'sssdp' from snap ssdp (0.0.1)
command 'sudo' from deb sudo
command 'sudo' from deb sudo-ldap
command 'sfdp' from deb graphviz
command 'sup' from deb sup
See 'snap info <snapname>' for additional versions.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -v -n
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:01:45.107922 IP (tos 0x0, ttl 1, id 32932, offset 0, flags [none], proto UDP (17), length 203)
192.168.0.194.65406 > 239.255.255.250.1900: UDP, length 175
11:01:45.431136 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.240 tell 192.168.0.107, length 46
11:01:45.566252 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.141 tell 192.168.0.190, length 46
11:01:45.590670 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.164 tell 192.168.0.107, length 46
11:01:45.738253 IP (tos 0x0, ttl 1, id 52371, offset 0, flags [none], proto UDP (17), length 204)
192.168.0.190.54153 > 239.255.255.250.1900: UDP, length 176
11:01:46.086093 IP (tos 0x0, ttl 1, id 29968, offset 0, flags [none], proto UDP (17), length 203)
192.168.0.166.65144 > 239.255.255.250.1900: UDP, length 175
11:01:46.097290 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.240 tell 192.168.0.107, length 46
11:01:46.108753 IP (tos 0x0, ttl 1, id 32933, offset 0, flags [none], proto UDP (17), length 203)
192.168.0.194.65406 > 239.255.255.250.1900: UDP, length 175
11:01:46.524893 IP6 (hlim 1, next-header UDP (17) payload length: 103) fe80::98b4:47fb:4996:5056.546 > ff02::1:2.547: [udp sum ok] dhcp6 solicit (xid=c0f377 (elapsed-time 6303) (client-ID hwaddr/time type 1 time 744492727 040e3c19288f) (IA_NA IAID:50597436 T1:0 T2:0) (Client-FQDN) (vendor-class) (option-request DNS-search-list DNS-server vendor-specific-info Client-FQDN))
11:01:46.566009 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.141 tell 192.168.0.190, length 46
11:01:47.046890 00:9e:1e:15:44:53 > 34:db:fd:77:e4:61, ethertype Unknown (0xa0a0), length 60:
0x0000: 0003 0101 0101 0101 0101 0101 0101 0101 ....
0x0010: 0101 0101 0101 0101 0101 0101 0101 0101 ....
0x0020: 0101 0101 0101 0101 0101 0101 ....
11:01:47.094578 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.240 tell 192.168.0.107, length 46
11:01:47.096305 IP (tos 0x0, ttl 1, id 29969, offset 0, flags [none], proto UDP (17), length 203)

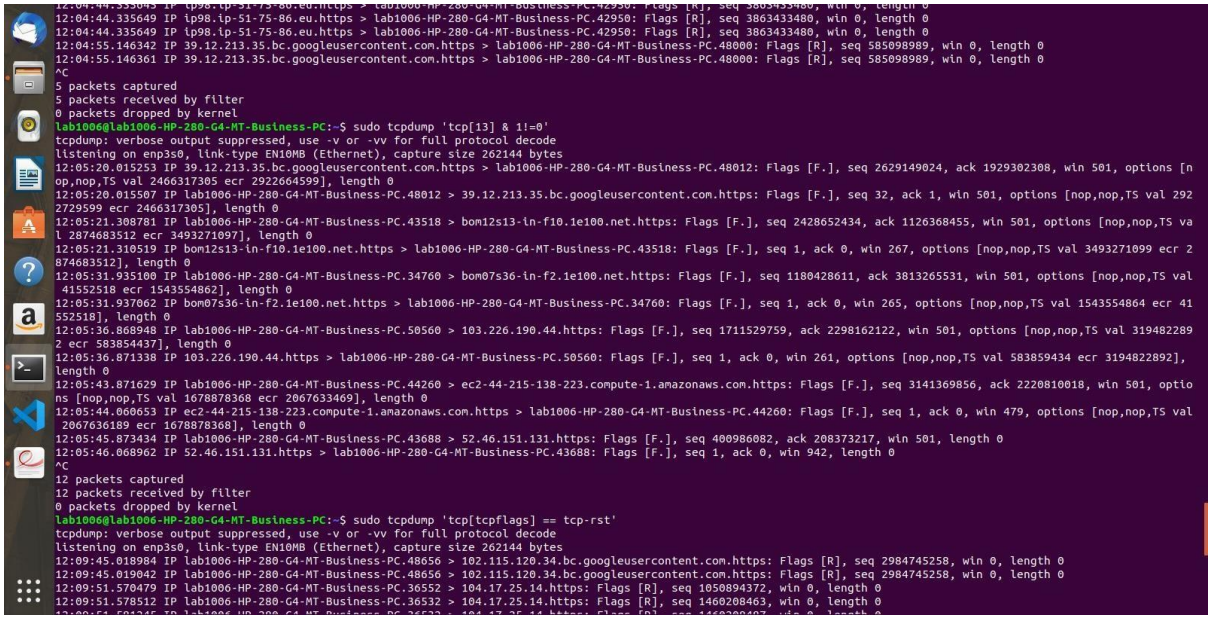

192.168.0.115.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
11:06:48.559410 04:0e:3c:19:2d:d2 > 33:33:00:00:00:fb, ethertype IPv6 (0x86dd), length 74: (hlim 1, next-header UDP (17) payload length: 20) fe80::d08:56ec:5b45:e946.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/0/0 (12)
11:06:48.559669 04:0e:3c:19:2d:d2 > 01:00:5e:00:00:fb, ethertype IPv4 (0x0800), length 60: (tos 0x0, ttl 1, id 17398, offset 0, flags [none], proto UDP (17), length 40)
192.168.0.148.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
11:06:48.859888 a4:ae:12:84:80:ea > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.134 tell 192.168.0.185, length 46
11:06:49.156505 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 218: (tos 0x0, ttl 1, id 620, offset 0, flags [none], proto UDP (17), length 204)
192.44.44.202.60046 > 239.255.255.250.1900: UDP, length 176
11:06:49.943390 04:0e:3c:1a:60:2f > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.141 tell 192.168.0.190, length 46
11:06:50.170534 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 218: (tos 0x0, ttl 1, id 621, offset 0, flags [none], proto UDP (17), length 204)
192.44.44.202.60046 > 239.255.255.250.1900: UDP, length 176
11:06:50.567535 04:0e:3c:1a:60:2f > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.141 tell 192.168.0.190, length 46
11:06:50.584752 04:0e:3c:1b:d1:42 > ac:15:a2:b9:9e:29, ethertype IPv4 (0x0800), length 105: (tos 0x0, ttl 64, id 15809, offset 0, flags [DF], proto TCP (6), length 91)
192.168.0.213.51252 > 185.199.108.154.443: Flags [P.], cksum 0xe82c (incorrect -> 0xc33f), seq 1873020008:1873020047, ack 1011178678, win 4607, options [nop,nop,TS val 3555857482 ecr 1680001507], length 39
11:06:50.601299 ac:15:a2:b9:9e:29 > 04:0e:3c:1b:d1:42, ethertype IPv4 (0x0800), length 66: (tos 0x34, ttl 55, id 60381, offset 0, flags [DF], proto TCP (6), length 52)
185.199.108.154.443 > 192.168.0.213.51252: Flags [.], cksum 0xbab0 (correct), ack 39, win 284, options [nop,nop,TS val 1680060306 ecr 3555857482], length 0
11:06:50.601323 ac:15:a2:b9:9e:29 > 04:0e:3c:1b:d1:42, ethertype IPv4 (0x0800), length 105: (tos 0x34, ttl 55, id 60382, offset 0, flags [DF], proto TCP (6), length 91)
185.199.108.154.443 > 192.168.0.213.51252: Flags [P.], cksum 0x5056 (correct), seq 1:40, ack 39, win 284, options [nop,nop,TS val 1680060306 ecr 3555857482], length 39
11:06:50.601400 04:0e:3c:1b:d1:42 > ac:15:a2:b9:9e:29, ethertype IPv4 (0x0800), length 66: (tos 0x0, ttl 64, id 15810, offset 0, flags [DF], proto TCP (6), length 52)
192.168.0.213.51252 > 185.199.108.154.443: Flags [.], cksum 0xe805 (incorrect -> 0xa995), ack 40, win 4607, options [nop,nop,TS val 3555857499 ecr 1680060306], length 0
11:06:50.675239 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 428: (tos 0x0, ttl 2, id 37639, offset 0, flags [DF], proto UDP (17), length 414)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 386
11:06:50.675463 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 437: (tos 0x0, ttl 2, id 37640, offset 0, flags [DF], proto UDP (17), length 423)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 395
11:06:50.675567 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 500: (tos 0x0, ttl 2, id 37641, offset 0, flags [DF], proto UDP (17), length 486)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 458
11:06:50.675836 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 496: (tos 0x0, ttl 2, id 37642, offset 0, flags [DF], proto UDP (17), length 482)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 454
11:06:50.675997 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 476: (tos 0x0, ttl 2, id 37643, offset 0, flags [DF], proto UDP (17), length 462)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 434
11:06:50.676187 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 508: (tos 0x0, ttl 2, id 37644, offset 0, flags [DF], proto UDP (17), length 494)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 466
11:06:50.676299 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 490: (tos 0x0, ttl 2, id 37645, offset 0, flags [DF], proto UDP (17), length 476)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 448
11:06:50.676493 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 492: (tos 0x0, ttl 2, id 37646, offset 0, flags [DF], proto UDP (17), length 478)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 450
11:06:50.676669 ac:15:a2:b9:9e:29 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 492: (tos 0x0, ttl 2, id 37647, offset 0, flags [DF], proto UDP (17), length 478)
192.168.0.1.60033 > 239.255.255.250.1900: UDP, length 450
11:06:50.696063 04:0e:3c:1a:5c:1f > 33:33:00:00:00:fb, ethertype IPv6 (0x86dd), length 102: (flowlabel 0x061d2, hlim 255, next-header UDP (17) payload length: 48) fe80:

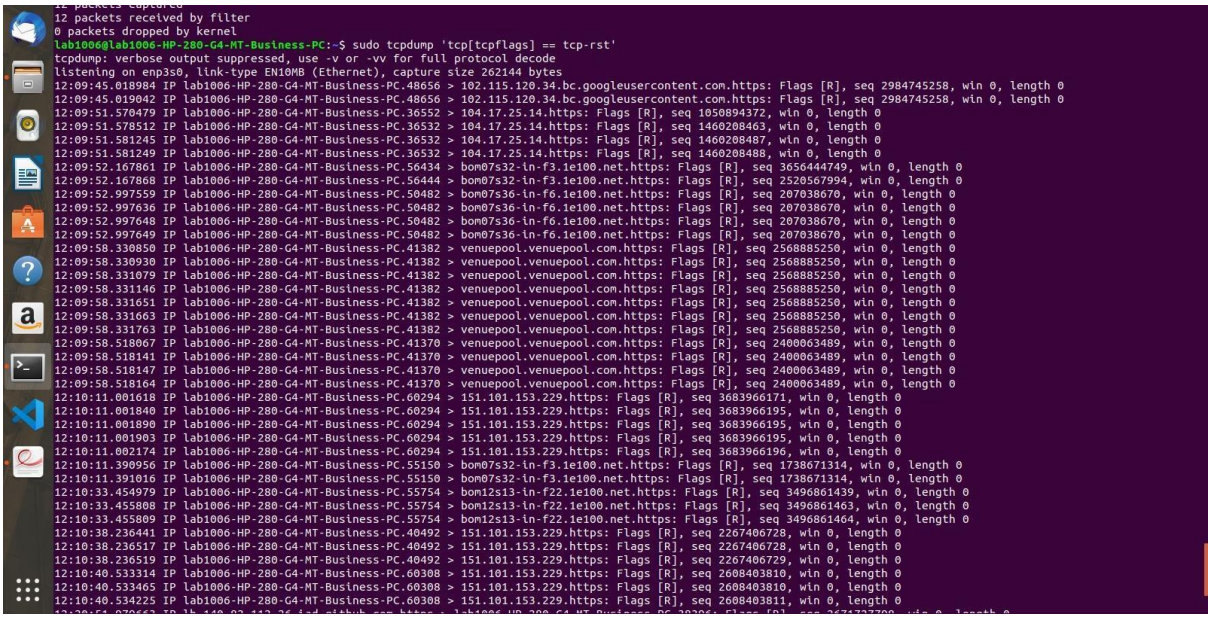

192.168.0.213.38292 > 152.195.38.76.80: Flags [.], cksum 0x80b3 (incorrect -> 0xbd9d), ack 1018572014, win 501, options [nop,nop,TS val 1531651325 ecr 4184076819], length 0
11:06:51.426290 ac:15:a2:b9:9e:29 > 04:0e:3c:1b:d1:42, ethertype IPv4 (0x0800), length 66: (tos 0x0, ttl 58, id 61638, offset 0, flags [none], proto TCP (6), length 52)
152.195.38.76.80 > 192.168.0.213.38292: Flags [.], cksum 0xeb9a (correct), ack 1, win 135, options [nop,nop,TS val 4184087059 ecr 1531629677], length 0
11:06:51.567263 04:0e:3c:1a:60:2f > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.141 tell 192.168.0.190, length 46
11:06:51.691627 a4:ae:12:84:80:ea > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.134 tell 192.168.0.185, length 46
11:06:51.831366 ac:15:a2:b9:9e:29 > 04:0e:3c:1b:d1:42, ethertype IPv4 (0x0800), length 91: (tos 0x34, ttl 46, id 7178, offset 0, flags [DF], proto TCP (6), length 77)
140.82.112.25.443 > 192.168.0.213.37992: Flags [P.], cksum 0xdcb4 (correct), seq 3601914876:3601914901, ack 4276782204, win 77, options [nop,nop,TS val 3554971416 ecr 3093326561], length 25
11:06:51.831411 04:0e:3c:1b:d1:42 > ac:15:a2:b9:9e:29, ethertype IPv4 (0x0800), length 66: (tos 0x0, ttl 64, id 46494, offset 0, flags [DF], proto TCP (6), length 52)
192.168.0.213.37992 > 140.82.112.25.443: Flags [.], cksum 0xbe0f (incorrect -> 0xc80d), ack 25, win 501, options [nop,nop,TS val 3093366561 ecr 3554971416], length 0
11:06:51.831638 04:0e:3c:1b:d1:42 > ac:15:a2:b9:9e:29, ethertype IPv4 (0x0800), length 95: (tos 0x0, ttl 64, id 46495, offset 0, flags [DF], proto TCP (6), length 81)
192.168.0.213.37992 > 140.82.112.25.443: Flags [P.], cksum 0xbe2c (incorrect -> 0x50e9), seq 1:30, ack 25, win 501, options [nop,nop,TS val 3093366561 ecr 3554971416], length 29
^C
43 packets captured
43 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n tcp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:08:51.831101 IP 140.82.112.25.443 > 192.168.0.213.37992: Flags [P.], seq 3601914926:3601914951, ack 4276782262, win 77, options [nop,nop,TS val 3555091411 ecr 3093426561], length 25
11:08:51.831308 IP 192.168.0.213.37992 > 140.82.112.25.443: Flags [P.], seq 1:30, ack 25, win 501, options [nop,nop,TS val 3093486561 ecr 3555091411], length 29
11:08:52.119414 IP 140.82.112.25.443 > 192.168.0.213.37992: Flags [.], ack 30, win 77, options [nop,nop,TS val 3555091700 ecr 3093486561], length 0
11:08:57.438567 IP 192.168.0.213.39510 > 140.82.114.22.443: Flags [.], seq 2871837009:2871837045, ack 1471998315, win 501, options [nop,nop,TS val 3554701205 ecr 242720383], length 36
11:08:57.438586 IP 192.168.0.213.39510 > 140.82.114.22.443: Flags [P.], seq 36:39, ack 1, win 501, options [nop,nop,TS val 3554701205 ecr 242720383], length 3
11:08:57.438660 IP 192.168.0.213.51198 > 185.199.108.154.443: Flags [.], seq 4064335366:4064335402, ack 768197512, win 11904, options [nop,nop,TS val 3555984336 ecr 1305689667], length 36
11:08:57.438664 IP 192.168.0.213.51198 > 185.199.108.154.443: Flags [P.], seq 36:39, ack 1, win 11904, options [nop,nop,TS val 3555984336 ecr 1305689667], length 3
11:08:57.439030 IP 192.168.0.213.51198 > 185.199.108.154.443: Flags [P.], seq 39:63, ack 1, win 11904, options [nop,nop,TS val 3555984337 ecr 1305689667], length 24
11:08:57.439057 IP 192.168.0.213.51198 > 185.199.108.154.443: Flags [F.], seq 63, ack 1, win 11904, options [nop,nop,TS val 3555984337 ecr 1305689667], length 0
11:08:57.439775 IP 192.168.0.213.39510 > 140.82.114.22.443: Flags [P.], seq 39:63, ack 1, win 501, options [nop,nop,TS val 3554701206 ecr 242720383], length 24
11:08:57.439800 IP 192.168.0.213.39510 > 140.82.114.22.443: Flags [F.], seq 63, ack 1, win 501, options [nop,nop,TS val 3554701206 ecr 242720383], length 0
11:08:57.455029 IP 185.199.108.154.443 > 192.168.0.213.51198: Flags [.], ack 0, win 377, options [nop,nop,TS val 1305742675 ecr 3555931344,nop,nop,sack 1 {36:39}], length 0
11:08:57.455044 IP 185.199.108.154.443 > 192.168.0.213.51198: Flags [.], ack 39, win 377, options [nop,nop,TS val 1305742675 ecr 3555984336], length 0
11:08:57.457877 IP 185.199.108.154.443 > 192.168.0.213.51198: Flags [.], ack 63, win 377, options [nop,nop,TS val 1305742675 ecr 3555984337], length 0
11:08:57.457892 IP 185.199.108.154.443 > 192.168.0.213.51198: Flags [P.], seq 1:25, ack 63, win 377, options [nop,nop,TS val 1305742675 ecr 3555984337], length 24
11:08:57.457894 IP 185.199.108.154.443 > 192.168.0.213.51198: Flags [F.], seq 25, ack 63, win 377, options [nop,nop,TS val 1305742675 ecr 3555984337], length 0
11:08:57.457958 IP 192.168.0.213.51198 > 185.199.108.154.443: Flags [R], seq 4064335429, win 0, length 0


25 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n tcp src 192.168.0.181
tcpdump: 'tcp' modifier applied to host
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n src 192.168.0.181 icmp
tcpdump: syntax error in filter expression: syntax error
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n icmp src 192.168.0.181 icmp
tcpdump: 'icmp' modifier applied to host
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:23:14.623598 IP 192.168.0.213 > 103.246.224.160: ICMP echo request, id 13898, seq 10, length 64
11:23:14.624221 IP 103.246.224.160 > 192.168.0.213: ICMP echo reply, id 13898, seq 10, length 64
11:23:15.647605 IP 192.168.0.213 > 103.246.224.160: ICMP echo request, id 13898, seq 11, length 64
11:23:15.648227 IP 103.246.224.160 > 192.168.0.213: ICMP echo reply, id 13898, seq 11, length 64
11:23:16.671565 IP 192.168.0.213 > 103.246.224.160: ICMP echo request, id 13898, seq 12, length 64
11:23:16.672192 IP 103.246.224.160 > 192.168.0.213: ICMP echo reply, id 13898, seq 12, length 64
11:23:17.695594 IP 192.168.0.213 > 103.246.224.160: ICMP echo request, id 13898, seq 13, length 64
11:23:17.696161 IP 103.246.224.160 > 192.168.0.213: ICMP echo reply, id 13898, seq 13, length 64
11:23:18.719632 IP 192.168.0.213 > 103.246.224.160: ICMP echo request, id 13898, seq 14, length 64
11:23:18.720145 IP 103.246.224.160 > 192.168.0.213: ICMP echo reply, id 13898, seq 14, length 64
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcp port 80
sudo: tcp: command not found
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:28:38.285039 IP lab1006-HP-280-G4-MT-Business-PC.49306 > 32.121.122.34.bc.googleusercontent.com.http: Flags [S], seq 3903811227, win 64240, options [mss 1460,sackOK,TS val 3444133253 ecr 0,nop,wscale 7], length 0
11:28:39.295561 IP lab1006-HP-280-G4-MT-Business-PC.49306 > 32.121.122.34.bc.googleusercontent.com.http: Flags [S], seq 3903811227, win 64240, options [mss 1460,sackOK,TS val 3444134263 ecr 0,nop,wscale 7], length 0
11:28:39.538360 IP 32.121.122.34.bc.googleusercontent.com.http > lab1006-HP-280-G4-MT-Business-PC.49306: Flags [S.], seq 1089476767, ack 3903811228, win 64768, options [mss 1420,sackOK,TS val 1089564564 ecr 3444134263,nop,wscale 7], length 0
11:28:39.538421 IP lab1006-HP-280-G4-MT-Business-PC.49306 > 32.121.122.34.bc.googleusercontent.com.http: Flags [.], ack 1, win 502, options [nop,nop,TS val 3444134506 ecr 1089564564], length 0


11:28:39.941579 IP 32.121.122.34.bc.googleusercontent.com.http > lab1006-HP-280-G4-MT-Business-PC.49306: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 1089565016 ecr 3444134506], length 0
11:28:39.941608 IP lab1006-HP-280-G4-MT-Business-PC.49306 > 32.121.122.34.bc.googleusercontent.com.http: Flags [.], ack 150, win 501, options [nop,nop,TS val 3444134909 ecr 1089565016], length 0
11:28:40.183386 IP 32.121.122.34.bc.googleusercontent.com.http > lab1006-HP-280-G4-MT-Business-PC.49306: Flags [.], ack 89, win 506, options [nop,nop,TS val 1089565258 ecr 3444134908], length 0
^C
12 packets captured
12 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump udp and src port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:33:37.241511 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.57215: 10986 9/3/1 A 34.122.121.32, A 35.224.170.84, A 185.125.190.18, A 35.232.111.17, A 91.189.91.48, A 185.125.190.49, A 185.125.190.17, A 91.189.91.49, A 185.125.190.48 (266)
11:33:37.241594 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.32907: 3486 6/3/1 AAAA 2001:67c:1562::23, AAAA 2620:2d:4000:1::23, AAAA 2620:2d:4000:1::2b, AAAA 2620:2d:4000:1::22, AAAA 2001:67c:1562::24, AAAA 2620:2d:4000:1::2a (290)
11:34:04.686194 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.53528: 54238 4/4/1 A 108.158.61.90, A 108.158.61.4, A 108.158.61.10, A 108.158.61.13 (258)
11:34:04.709453 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.59252: 37086 8/4/1 AAAA 2600:9000:237b:c200:1a:5235:f980:93a1, AAAA 2600:9000:237b:c000:1a:5235:f980:93a1, AAAA 2600:9000:237b:400:1a:5235:f980:93a1, AAAA 2600:9000:237b:d400:1a:5235:f980:93a1, AAAA 2600:9000:237b:7800:1a:5235:f980:93a1, AAAA 2600:9000:237b:7e00:1a:5235:f980:93a1, AAAA 2600:9000:237b:dc00:1a:5235:f980:93a1, AAAA 2600:9000:237b:2800:1a:5235:f980:93a1 (418)
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump portrange 1-80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:35:13.653873 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74 (oui Unknown), length 300
11:35:17.801654 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74 (oui Unknown), length 300
11:35:22.173999 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74 (oui Unknown), length 300
11:35:30.078393 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74 (oui Unknown), length 300
11:35:38.922635 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 1c:6f:65:ae:98:2a (oui Unknown), length 300
11:35:41.918550 IP lab1006-HP-280-G4-MT-Business-PC.36586 > _gateway.domain: 53847+ [1au] A? encrypted-tbn0.gstatic.com. (55)
11:35:41.918818 IP lab1006-HP-280-G4-MT-Business-PC.35381 > _gateway.domain: 12276+ [1au] AAAA? encrypted-tbn0.gstatic.com. (55)
11:35:41.919849 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.36586: 53847 1/0/1 A 142.250.183.78 (71)
11:35:41.938280 IP lab1006-HP-280-G4-MT-Business-PC.56668 > _gateway.domain: 933+ [1au] A? www.google.com. (43)
11:35:41.938421 IP lab1006-HP-280-G4-MT-Business-PC.59077 > _gateway.domain: 26727+ [1au] AAAA? www.google.com. (43)
11:35:41.939510 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.56668: 933 1/0/1 A 172.217.27.196 (59)
11:35:41.939601 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.59077: 26727 1/0/1 AAAA 2404:6800:4009:800::2004 (71)
11:35:41.980589 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.35381: 12276 1/0/1 AAAA 2404:6800:4009:822::200e (83)
11:35:42.677951 IP lab1006-HP-280-G4-MT-Business-PC.37545 > _gateway.domain: 56141+ [1au] A? www.gstatic.com. (44)
11:35:42.678020 IP lab1006-HP-280-G4-MT-Business-PC.41726 > _gateway.domain: 30891+ [1au] AAAA? www.gstatic.com. (44)
11:35:42.679208 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.41726: 30891 1/0/1 AAAA 2404:6800:4009:82b::2003 (72)
11:35:42.679329 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.37545: 56141 1/0/1 A 142.250.192.131 (60)


11:35:42.744434 IP lab1006-HP-280-G4-MT-Business-PC.55375 > _gateway.domain: 35292+ [1au] A? apis.google.com. (44)
11:35:42.744508 IP lab1006-HP-280-G4-MT-Business-PC.47736 > _gateway.domain: 45730+ [1au] AAAA? apis.google.com. (44)
11:35:42.745662 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.55375: 35292 2/0/1 CNAME plus.l.google.com., A 142.251.42.78 (81)
11:35:42.745668 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.47736: 45730 2/0/1 CNAME plus.l.google.com., AAAA 2404:6800:4009:831::200e (93)
11:35:42.845172 IP lab1006-HP-280-G4-MT-Business-PC.55210 > _gateway.domain: 48143+ [1au] A? adservice.google.com. (49)
11:35:42.845258 IP lab1006-HP-280-G4-MT-Business-PC.51043 > _gateway.domain: 27592+ [1au] AAAA? adservice.google.com. (49)
11:35:42.846395 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.55210: 48143 1/0/1 A 142.250.192.98 (65)
11:35:42.846733 IP lab1006-HP-280-G4-MT-Business-PC.39669 > _gateway.domain: 31162+ [1au] A? safebrowsing.googleapis.com. (56)
11:35:42.846788 IP lab1006-HP-280-G4-MT-Business-PC.48992 > _gateway.domain: 63325+ [1au] AAAA? safebrowsing.googleapis.com. (56)
11:35:42.847885 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.48992: 63325 1/0/1 AAAA 2404:6800:4009:823::200a (84)
11:35:42.847898 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.39669: 31162 1/0/1 A 142.250.183.106 (72)
11:35:42.850258 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.51043: 27592 1/0/1 AAAA 2404:6800:4009:820::2002 (77)
11:35:43.014836 IP lab1006-HP-280-G4-MT-Business-PC.43491 > _gateway.domain: 41945+ [1au] A? adservice.google.co.in. (51)
11:35:43.014910 IP lab1006-HP-280-G4-MT-Business-PC.35711 > _gateway.domain: 33071+ [1au] AAAA? adservice.google.co.in. (51)
11:35:43.015190 IP lab1006-HP-280-G4-MT-Business-PC.54633 > _gateway.domain: 59138+ [1au] A? googleads.g.doubleclick.net. (56)
11:35:43.015251 IP lab1006-HP-280-G4-MT-Business-PC.34413 > _gateway.domain: 1087+ [1au] AAAA? googleads.g.doubleclick.net. (56)
11:35:43.016017 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.43491: 41945 2/0/1 CNAME pagead46.l.doubleclick.net., A 142.250.192.34 (107)
11:35:43.016055 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.35711: 33071 2/0/1 CNAME pagead46.l.doubleclick.net., AAAA 2404:6800:4009:823::2002 (119)
11:35:43.016261 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.54633: 59138 1/0/1 A 142.250.199.130 (72)
11:35:43.039586 IP _gateway.domain > lab1006-HP-280-G4-MT-Business-PC.34413: 1087 1/0/1 AAAA 2404:6800:4009:82c::2002 (84)
11:35:45.136757 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74 (oui Unknown), length 300
^C
38 packets captured
38 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump port 80 -w capture_1
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C86 packets captured
86 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -nnvvS src 10.5.2.3 and dst port 3389
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -nnvvS src 103.246.224.160 and dst port 3389
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ tcpdump 'tcp[13] & 32!=0'
tcpdump: enp3s0: You don't have permission to capture on that device
(socket: Operation not permitted)





CONCLUSION:

We gained a practical understanding of how TCPDump can be employed to capture, dissect, and interpret network packets in real-time, offering valuable insights into network behavior, troubleshooting, and security assessment. By applying various filters and commands, we were able to capture specific types of traffic based on source and destination addresses, protocols, ports, and packet sizes.

Assignment 8

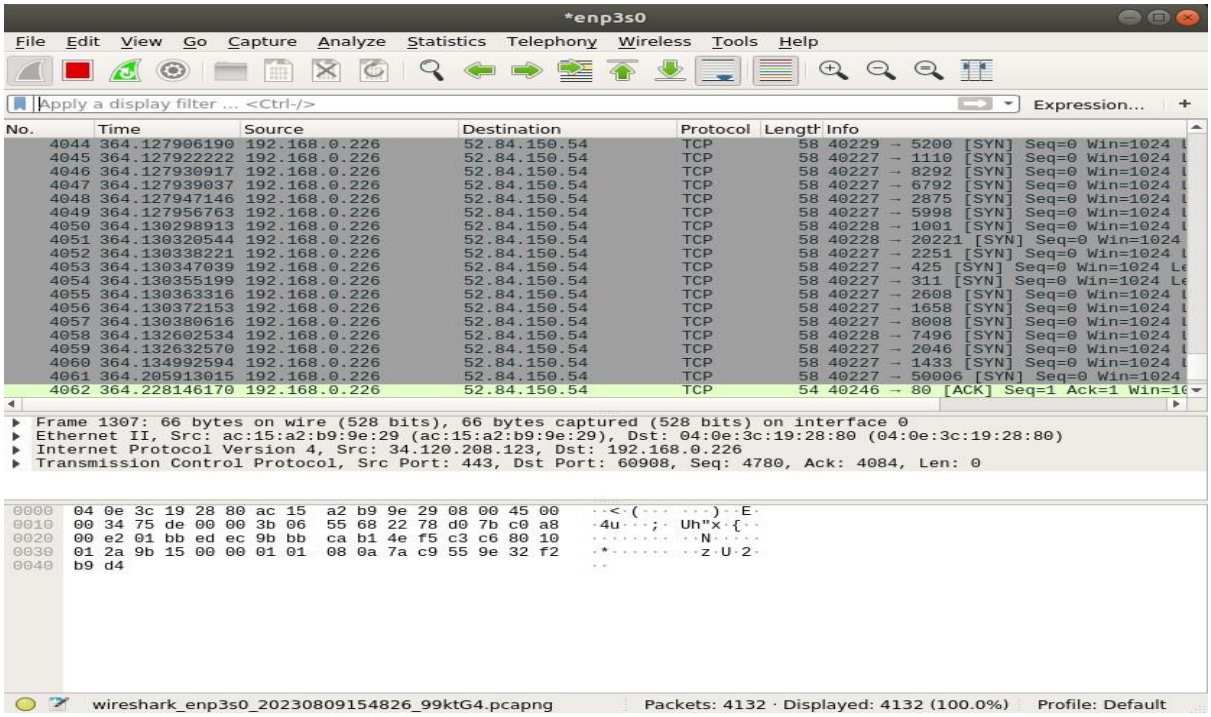
Aim:- Installation of NMAP and using it with different options to scan open ports, perform OS fingerprinting, ping scan, TCP port scan, etc

LO mapped: - LO4

Theory:-

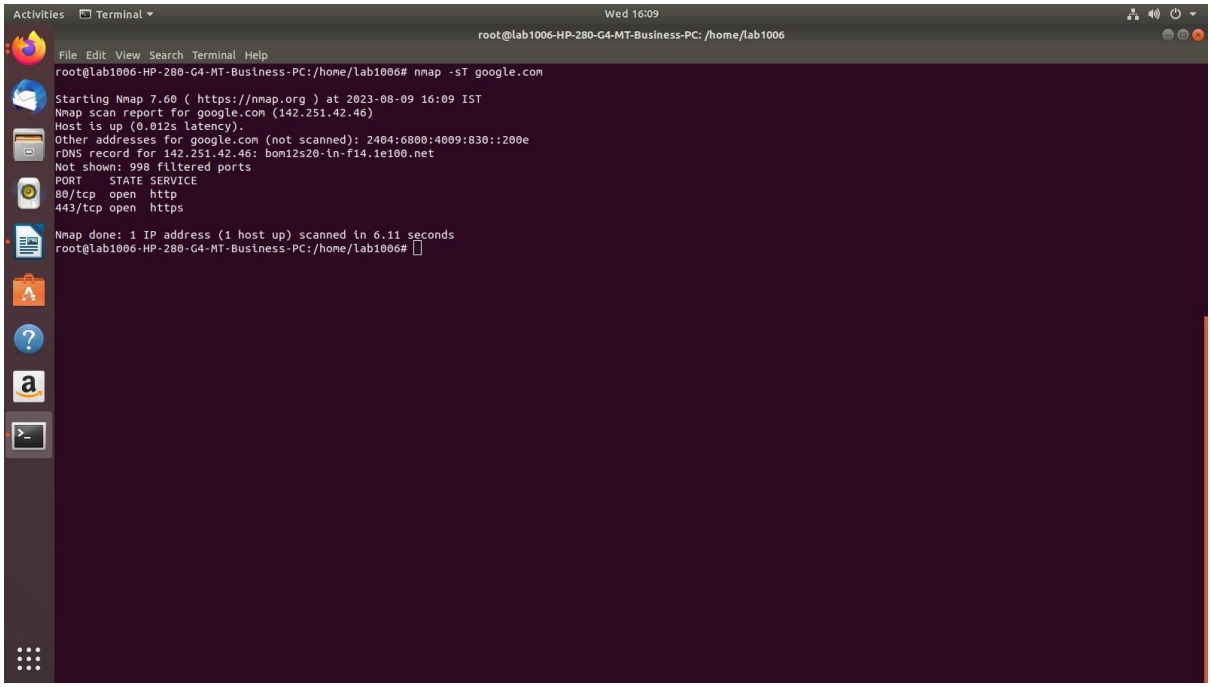
Ping Sweep
Nmap -sP <IP address(192.168.0.*)>

1. -sS (TCP SYN scan)



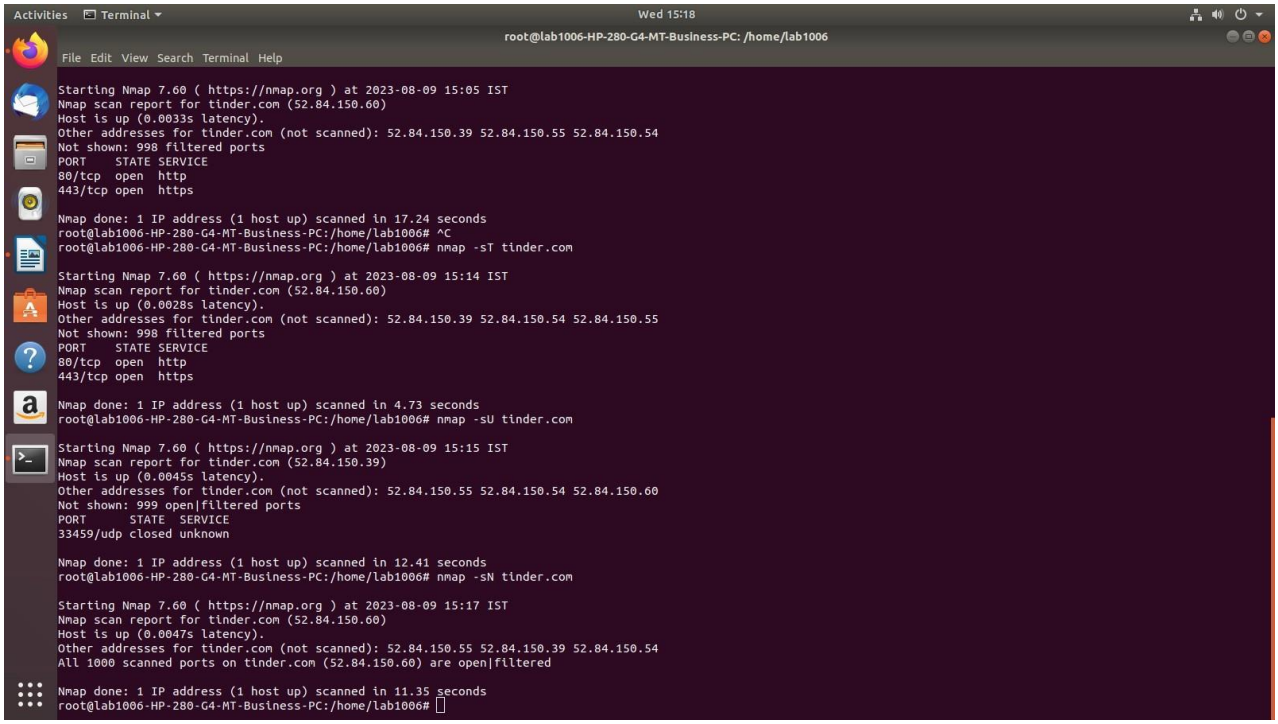
SYN scan is the default and most popular scan option for good reason. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by intrusive firewalls. SYN scan is relatively unobtrusive and stealthy, since it never completes TCP connections. It also works against any compliant TCP stack rather than depending on idiosyncrasies of specific platforms as Nmap's FIN/NULL/Xmas, Maimon and idle scans do. It also allows clear, reliable differentiation between open, closed, and filtered states

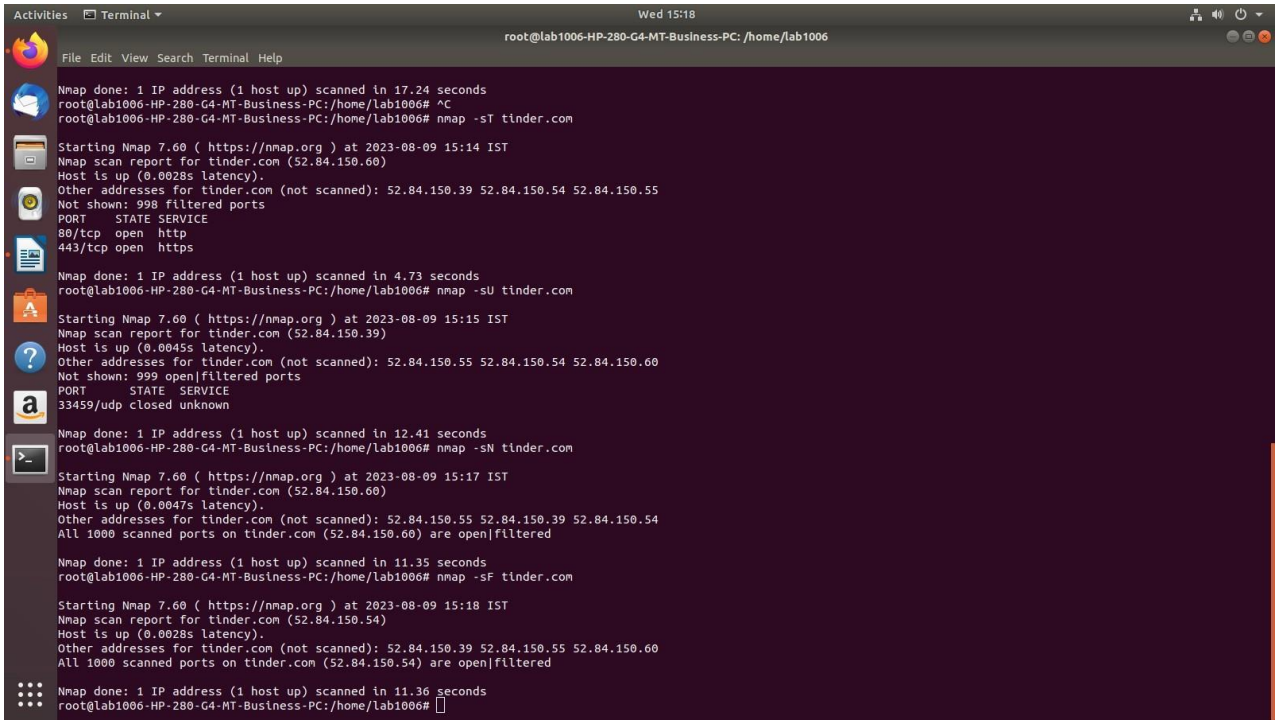
2. -sT (TCP connect scan)



TCP connect scan is the default TCP scan type when SYN scan is not an option. This is the case when a user does not have raw packet privileges or is scanning IPv6 networks. Instead of writing raw packets as most other scan types do, Nmap asks the underlying operating system to establish a connection with the target machine and port by issuing the connect system call. This is the same high-level system call that web browsers, P2P clients, and most other network-enabled applications use to establish a connection. It is part of a programming interface known as the Berkeley Sockets API. Rather than read raw packet responses off the wire, Nmap uses this API to obtain status information on each connection attempt. This and the FTP bounce scan (the section called "TCP FTP Bounce Scan (-b)") are the only scan types available to unprivileged users.

3. -sU (UDP scans)





These three scan types (even more are possible with the --scanflags option described in the next section) exploit a subtle loophole in the TCP RFC to differentiate between open and closed ports. Page 65 of RFC 793 says that "if the [destination] port state is CLOSED .... an incoming segment not containing a RST causes a RST to be sent in response." Then the next page discusses packets sent to open ports without the SYN, RST, or ACK bits set, stating that: "you are unlikely to get here, but if you do, drop the segment, and return."

When scanning systems compliant with this RFC text, any packet not containing SYN, RST, or ACK bits will result in a returned RST if the port is closed and no response at all if the port is open. As long as none of those three bits are included, any combination of the other three (FIN, PSH, and URG) are OK. Nmap exploits this with three scan types: Null scan (-sN)

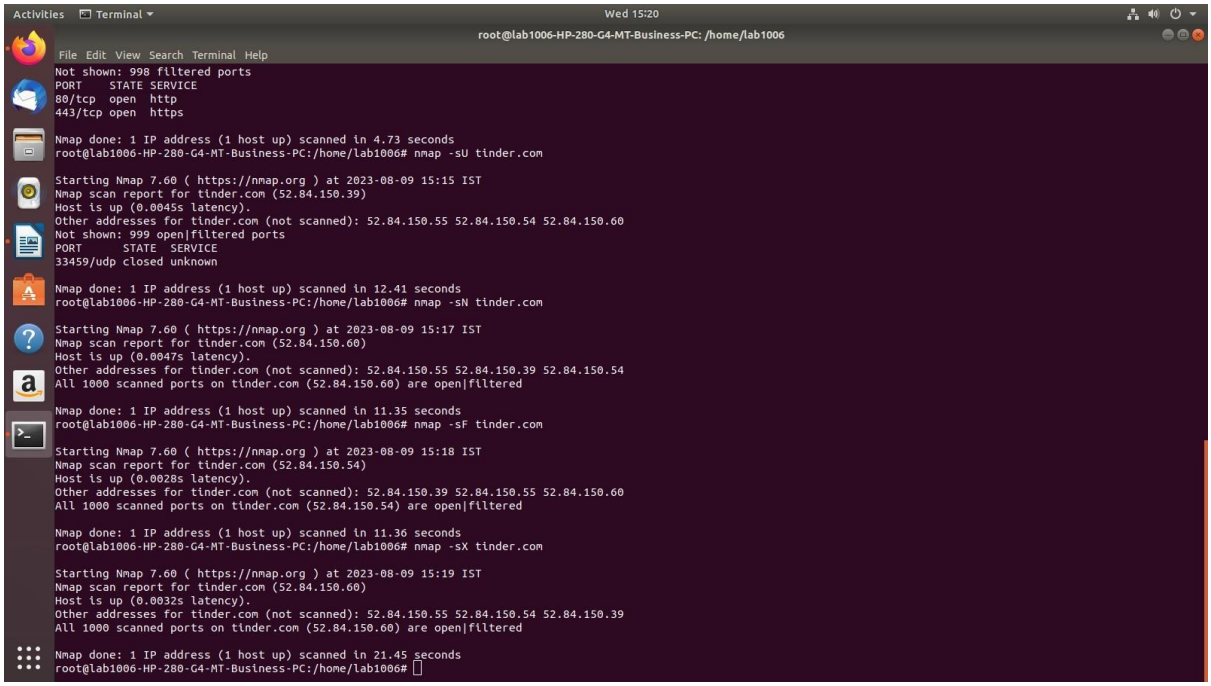
Does not set any bits (TCP flag header is 0)

FIN scan (-sF)

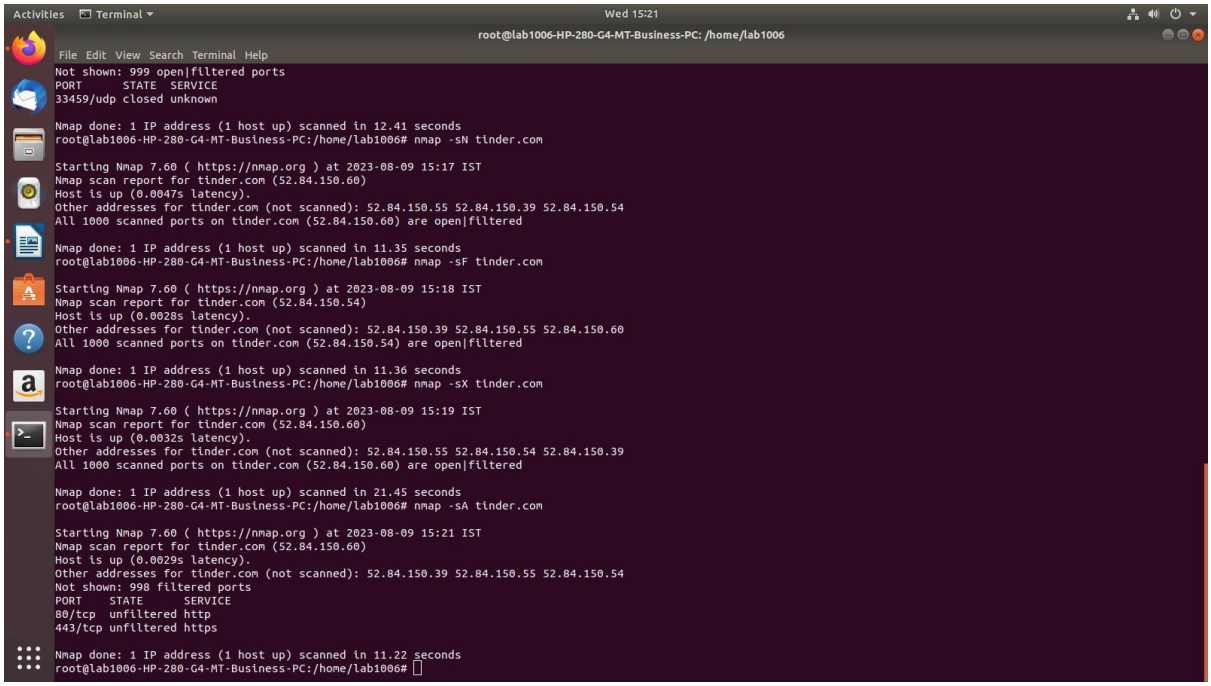
Sets just the TCP FIN bit.

Xmas scan (-sX)

Sets the FIN, PSH, and URG flags, lighting the packet up like a Christmas tree.



4-sA (TCP ACK scan)

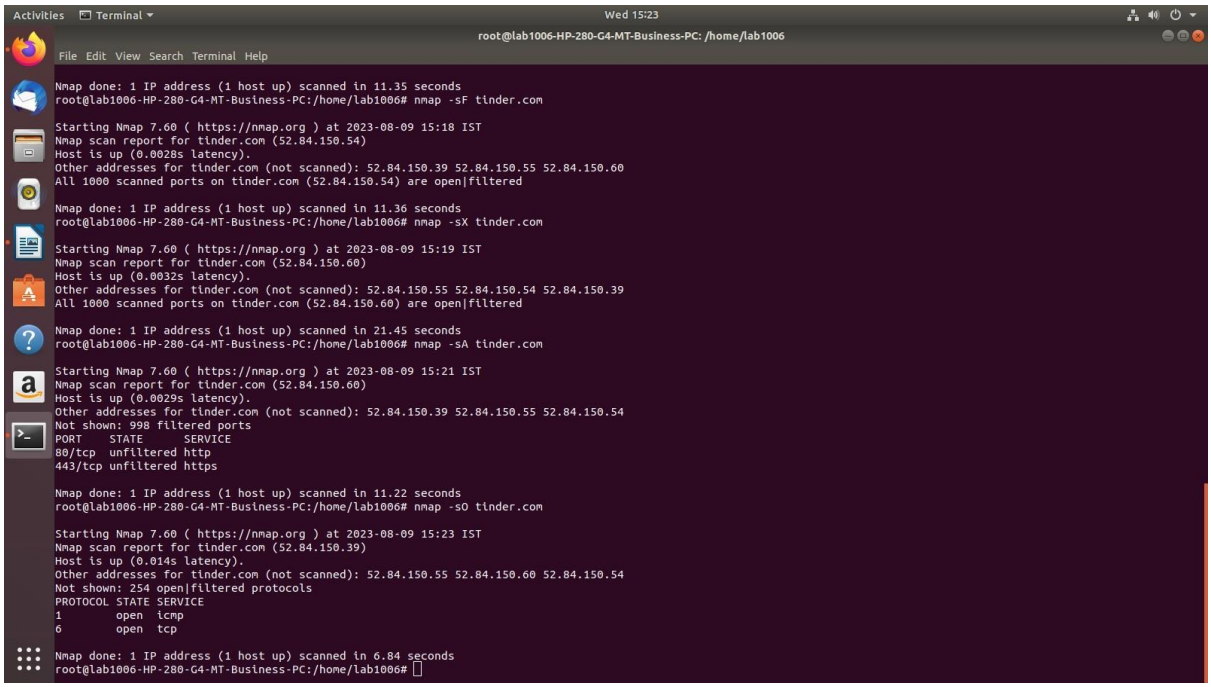


This scan is different than the others discussed so far in that it never determines open (or even open|filtered) ports. It is used to map out firewall rulesets, determining whether they are stateful or not and which ports are filtered.

ACK scan is enabled by specifying the -sA option. Its probe packet has only the ACK flag set (unless you use --scanflags). When scanning unfiltered systems, open and closed ports will both return a RST packet. Nmap then labels them as unfiltered, meaning that they are reachable by the ACK packet, but whether they are open or closed is

undetermined. Ports that don't respond, or send certain ICMP error messages back, are labeled filtered. Table 5.5 provides the full details.

5. -sO (IP protocol scan)



IP protocol scan allows you to determine which IP protocols (TCP, ICMP, IGMP, etc.) are supported by target machines. This isn't technically a port scan, since it cycles through IP protocol numbers rather than TCP or UDP port numbers. Yet it still uses the -p option to select scanned protocol numbers, reports its results within the normal port table format, and even uses the same underlying scan engine as the true port scanning methods. So it is close enough to a port scan that it belongs here.

Besides being useful in its own right, protocol scan demonstrates the power of open-source software. While the fundamental idea is pretty simple, I had not thought to add it nor received any requests for such functionality. Then in the summer of 2000, Gerhard Rieger conceived the idea, wrote an excellent patch implementing it, and sent it to the *nmap-hackers* mailing list. I incorporated that patch into the Nmap tree and released a new version the next day. Few pieces of commercial software have users enthusiastic enough to design and contribute their own improvements!

6 .-O (Enable OS detection)

```
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006# nmap -O 192.168.0.119

Starting Nmap 7.60 ( https://nmap.org ) at 2023-08-09 15:36 IST
Nmap scan report for 192.168.0.119
Host is up (0.00029s latency).
All 1000 scanned ports on 192.168.0.119 are closed
MAC Address: 04:0E:3C:1A:5F:16 (Unknown)
Too many fingerprints match this host to give specific OS details
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.66 seconds
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006#
```

One of Nmap's best-known features is remote OS detection using TCP/IP stack fingerprinting.

Nmap sends a series of TCP and UDP packets to the remote host and examines practically every bit in the responses. After performing dozens of tests such as TCP ISN sampling, TCP options support and ordering, IP ID sampling, and the initial window size check, Nmap compares the results to its nmap-os-db database of more than 2,600 known OS fingerprints and prints out the OS details if there is a match. Each fingerprint includes a freeform textual description of the OS, and a classification which provides the vendor name (e.g. Sun), underlying OS (e.g. Solaris), OS generation (e.g. 10), and device type (general purpose, router, switch, game console, etc). Most fingerprints also have a Common Platform Enumeration (CPE) representation, like cpe:/o:linux:linux_kernel:2.6.

7 nmap -sP 192.168.0.*

```
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006# nmap -sP 192.168.0.*

Starting Nmap 7.60 ( https://nmap.org ) at 2023-08-09 15:39 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.00042s latency).
MAC Address: AC:15:A2:B9:9E:29 (Unknown)
Nmap scan report for 192.168.0.105
Host is up (-0.100s latency).
MAC Address: A4:AE:12:84:7F:CF (Unknown)
Nmap scan report for 192.168.0.114
Host is up (-0.10s latency).
MAC Address: 04:0E:3C:19:2E:0F (Unknown)
Nmap scan report for 192.168.0.115
Host is up (-0.100s latency).
MAC Address: 04:0E:3C:1A:5C:AD (Unknown)
Nmap scan report for 192.168.0.116
Host is up (-0.100s latency).
MAC Address: 04:0E:3C:1A:60:A0 (Unknown)
Nmap scan report for 192.168.0.117
Host is up (-0.10s latency).
MAC Address: 04:0E:3C:19:2D:1C (Unknown)
Nmap scan report for 192.168.0.118
Host is up (0.00080s latency).
MAC Address: E4:54:E8:C6:37:76 (Unknown)
Nmap scan report for 192.168.0.119
Host is up (0.00020s latency).
MAC Address: 04:0E:3C:1A:5F:16 (Unknown)
Nmap scan report for 192.168.0.121
Host is up (-0.099s latency).
MAC Address: 90:8D:78:7E:5A:B3 (D-Link International)
Nmap scan report for 192.168.0.123
Host is up (-0.100s latency).
MAC Address: F4:39:09:49:0A:33 (Unknown)
Nmap scan report for 192.168.0.126
Host is up (-0.10s latency).
MAC Address: 04:0E:3C:1A:61:7F (Unknown)
Nmap scan report for 192.168.0.133
Host is up (-0.10s latency).
MAC Address: A0:8C:FD:C5:AD:A1 (Hewlett Packard)
Nmap scan report for 192.168.0.135
Host is up (-0.10s latency).
MAC Address: A0:8C:FD:DD:8C:AE (Hewlett Packard)
Nmap scan report for 192.168.0.141
Host is up (-0.100s latency).
```

A ping sweep (also known as an ICMP sweep) is a basic network scanning technique used to determine which of a range of IP addresses map to live hosts (computers).

Whereas a single ping will tell whether one specified host computer exists on the network, a ping sweep consists of ICMP (Internet Control Message Protocol) *echo requests* sent to multiple hosts. To do this, the ping requires an address to send the echo request to, which can be an IP address or a web server domain name.

If a given address is live, it will return an ICMP *echo reply*. To disable ping sweeps on a network, administrators can block ICMP *echo requests* from outside sources. However, ICMP *timestamp* and *Address Mask requests* can be used in a similar manner.

CONCLUSION :- By this assignment we implemented various different nmap network scanning commands and used wireshark.

Animesh Parab t2-t21 88

Assignment

Aim:- Simulate DOS attack using HPING3.

Lab Outcome Attained :- LO5

Theory:-

What is Denial of Service Atack?

A Denial of Service (DoS) attack is a malicious attempt to disrupt the normal functioning of a computer system, network, or online service by overwhelming it with a flood of illegitimate requests or traffic. The primary goal of a DoS attack is to make a resource, such as a website or server, unavailable to its intended users. It does so by consuming the target's resources, such as bandwidth, processing power, or memory, to the point where it cannot handle legitimate requests.

Explain SYN flood, ICMP flood and SMURF attack.

Three common types of DoS attacks:

SYN Flood Attack:

A SYN flood attack is a type of network-based DoS attack that targets the threeway handshake process in the Transmission Control Protocol (TCP), which is used for establishing connections between devices on the internet.

In a TCP connection, the client sends a SYN (synchronize) packet to initiate a connection with a server. The server is expected to respond with a SYN-ACK (synchronizeacknowledgment) packet, and then the client responds with an ACK

(acknowledgment) packet to complete the handshake and establish the connection. In a SYN flood attack, the attacker sends a high volume of SYN packets to the target server, but they do not complete the handshake by sending the expected ACK packets. This leaves the server waiting for the final ACKs, tying up its resources and preventing it from accepting legitimate connections.

SYN flood attacks can quickly overwhelm a server's ability to handle incoming connections, leading to service disruption.

ICMP Flood Attack:

An ICMP (Internet Control Message Protocol) flood attack, also known as a "ping flood" attack, targets the ICMP protocol, which is used for network diagnostics, particularly the "ping" command.

In this type of attack, the attacker sends a high volume of ICMP echo requests (ping requests) to the target system. Each request typically generates a response from the target, creating a flood of traffic.

ICMP flood attacks can consume the target's network bandwidth and processing resources, making it difficult for legitimate network traffic to pass through. This results in network congestion and service degradation or unavailability.

SMURF Attack:

A SMURF attack is a network-based DoS attack that takes advantage of ICMP and IP addressing.

In a SMURF attack, the attacker sends a large number of ICMP echo request (ping) packets to an IP broadcast address, typically spoofing the source IP address to make it appear as if the requests are coming from the victim's IP address. When these requests are sent to the broadcast address, all devices on the target network respond with ICMP echo replies. With a high enough volume of requests, this can flood the victim's network, overwhelming its resources and causing a DoS. To mitigate DoS attacks, organizations use various security measures, including firewalls, intrusion detection systems (IDS), intrusion prevention systems (IPS), and content delivery networks (CDNs). These tools help identify and filter out malicious traffic, allowing legitimate traffic to reach its destination. Additionally, proper network design and configuration can help minimize the impact of DoS attacks.

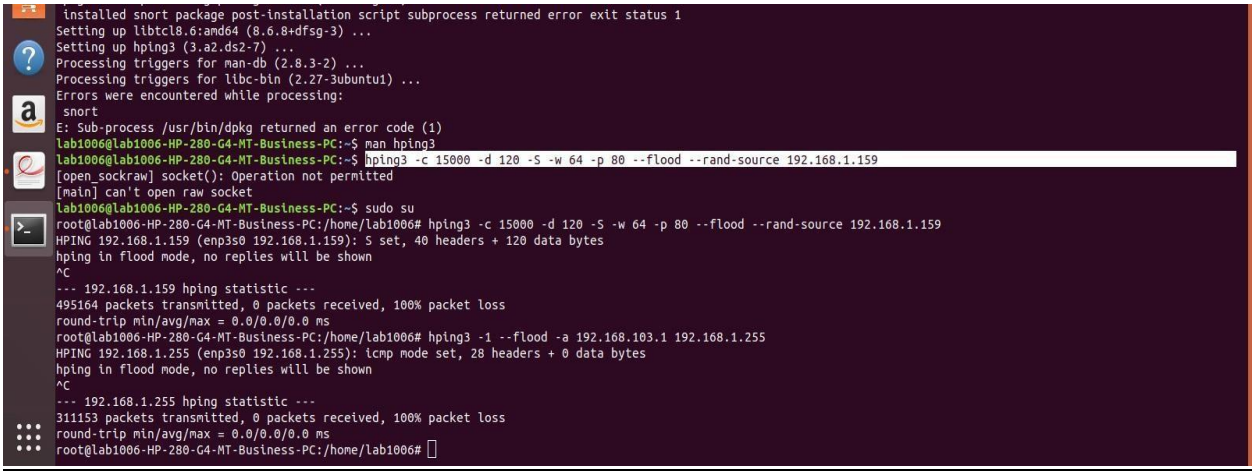
Write the Hping3 commands used for performing SYN flood and ICMP flood.

Syn flood :

hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159 ICMP flood:

hping3 -1 --flood -a 192.168.103 192.168.1.255

Output Screenshots:-



```
21:33:33.482317 IP 135.115.228.190.5553 > 192.168.1.159.80: Flags [S], seq 501438372:501438492, win 64, length 120: HTTP
21:33:33.487687 IP 246.196.86.246.5554 > 192.168.1.159.80: Flags [S], seq 1440614849:1440614969, win 64, length 120: HTTP
21:33:33.512837 IP 159.255.118.25.5555 > 192.168.1.159.80: Flags [S], seq 525299087:525299127, win 64, length 120: HTTP
21:33:33.522520 IP 159.157.124.107.5556 > 192.168.1.159.80: Flags [S], seq 1481118570:1481118690, win 64, length 120: HTTP
21:33:33.526511 IP 49.165.200.13.5765 > 192.168.1.159.80: Flags [S], seq 373764103:373764223, win 64, length 120: HTTP
21:33:33.533558 IP 117.137.227.248.5557 > 192.168.1.159.80: Flags [S], seq 1253057799:1253305919, win 64, length 120: HTTP
21:33:33.545209 IP 69.72.136.176.5558 > 192.168.1.159.80: Flags [S], seq 1703744130:1703744250, win 64, length 120: HTTP
21:33:33.545235 IP 105.4.221.89.5560 > 192.168.1.159.80: Flags [S], seq 121504987:121505107, win 64, length 120: HTTP
21:33:33.556586 IP 196.165.228.164.5561 > 192.168.1.159.80: Flags [S], seq 901964969:901965089, win 64, length 120: HTTP
21:33:33.563569 IP 227.152.5.127.5802 > 192.168.1.159.80: Flags [S], seq 846471944:846472064, win 64, length 120: HTTP
21:33:33.572473 IP 164.47.172.26.5562 > 192.168.1.159.80: Flags [S], seq 1145998142:1145998262, win 64, length 120: HTTP
21:33:33.579359 IP 22.29.2.52.5563 > 192.168.1.159.80: Flags [S], seq 169889196:169889316, win 64, length 120: HTTP
21:33:33.589088 IP 227.191.79.36.5608 > 192.168.1.159.80: Flags [S], seq 1244124856:1244124976, win 64, length 120: HTTP
21:33:33.603306 IP 119.227.36.233.5565 > 192.168.1.159.80: Flags [S], seq 438166914:438167034, win 64, length 120: HTTP
21:33:33.604965 IP 207.55.129.246.5566 > 192.168.1.159.80: Flags [S], seq 131638839:131638959, win 64, length 120: HTTP
21:33:33.622460 IP 250.227.48.248.5593 > 192.168.1.159.80: Flags [S], seq 1088210362:1088210482, win 64, length 120: HTTP
21:33:33.626866 IP 7.114.52.171.5568 > 192.168.1.159.80: Flags [S], seq 194261348:194261468, win 64, length 120: HTTP
21:33:33.632386 IP 186.120.2.86.5569 > 192.168.1.159.80: Flags [S], seq 1238701635:1238701755, win 64, length 120: HTTP
21:33:33.636133 IP 122.58.199.7.5571 > 192.168.1.159.80: Flags [S], seq 1637904461:1637904581, win 64, length 120: HTTP
21:33:33.638521 IP 47.189.125.148.5572 > 192.168.1.159.80: Flags [S], seq 1257911524:1257911644, win 64, length 120: HTTP
21:33:33.644830 IP 65.124.137.143.5583 > 192.168.1.159.80: Flags [S], seq 2132656423:2132656543, win 64, length 120: HTTP
21:33:33.652589 IP 140.135.114.239.5573 > 192.168.1.159.80: Flags [S], seq 13967979:13968099, win 64, length 120: HTTP
21:33:33.664973 IP 152.207.201.27.5775 > 192.168.1.159.80: Flags [S], seq 1387484552:1387484672, win 64, length 120: HTTP
21:33:33.686105 IP 54.118.229.124.5575 > 192.168.1.159.80: Flags [S], seq 1062466761:1062466881, win 64, length 120: HTTP
21:33:33.694483 IP 159.65.71.52.5576 > 192.168.1.159.80: Flags [S], seq 1663226321:1663226441, win 64, length 120: HTTP
21:33:33.712885 IP 201.199.60.73.5640 > 192.168.1.159.80: Flags [S], seq 129207029:129207149, win 64, length 120: HTTP
21:33:33.721387 IP 248.217.128.89.5577 > 192.168.1.159.80: Flags [S], seq 1695826100:1695826226, win 64, length 120: HTTP
21:33:33.728655 IP 69.131.161.248.5752 > 192.168.1.159.80: Flags [S], seq 173342584:173342704, win 64, length 120: HTTP
21:33:33.740884 IP 111.231.65.49.5578 > 192.168.1.159.80: Flags [S], seq 1957966395:1957966515, win 64, length 120: HTTP
21:33:33.742752 IP 193.221.190.198.5585 > 192.168.1.159.80: Flags [S], seq 1282735405:1282735525, win 64, length 120: HTTP
21:33:33.744834 IP 199.151.157.12.5579 > 192.168.1.159.80: Flags [S], seq 995146271:995146391, win 64, length 120: HTTP
21:33:33.756985 IP 55.186.160.25.5580 > 192.168.1.159.80: Flags [S], seq 218571662:218571782, win 64, length 120: HTTP
21:33:33.770517 IP 127.21.135.135.5581 > 192.168.1.159.80: Flags [S], seq 1963358298:1963358418, win 64, length 120: HTTP
21:33:33.778123 IP 47.71.231.2.5582 > 192.168.1.159.80: Flags [S], seq 1760895:1276670715, win 64, length 120: HTTP
21:33:33.778349 IP 64.105.171.116.5586 > 192.168.1.159.80: Flags [S], seq 3936787163936788836, win 64, length 120: HTTP
21:33:33.780154 IP 218.231.5.51.5589 > 192.168.1.159.80: Flags [S], seq 1444643947:1444644067, win 64, length 120: HTTP
21:33:33.782417 IP 0.159.158.54.5590 > 192.168.1.159.80: Flags [S], seq 363642928:363643048, win 64, length 120: HTTP
21:33:33.784283 IP 237.52.17.13.5591 > 192.168.1.159.80: Flags [S], seq 1807319071:1807319191, win 64, length 120: HTTP
21:33:33.797909 IP 231.65.217.180.5734 > 192.168.1.159.80: Flags [S], seq 411917354:411917474, win 64, length 120: HTTP
21:33:44.564540 IP6 fe80::e524:8e09:bf1b:b6b7 > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
21:33:44.645812 IP 0.0.0.68 > 255.255.255.255.67: BOOTP/DHCP, Request from 08:00:27:1a:eb:ad, length 300
21:33:44.848756 IP6 fe80::e524:8e09:bf1b:b6b7 > ff02::16: HBH ICMP6, multicast listener report v2, 2 group record(s), length 48
21:33:45.540538 ARP, Request who-has 10.0.2.2 tell 10.0.2.15, length 28
21:33:46.568282 ARP, Request who-has 10.0.2.2 tell 10.0.2.15, length 28
21:33:47.595714 ARP, Request who-has 10.0.2.2 tell 10.0.2.15, length 28
21:33:47.897631 IP 0.0.0.0.68 > 255.255.255.255.67: BOOTP/DHCP, Request from 08:00:27:1a:eb:ad, length 300
```

```
File  Edit  View  Search  Terminal  Help
21:33:54.823111 IP 37.93.65.14.45923 > 192.168.1.159.80: Flags [S], seq 1689138181:1689138301, win 64, length 120: HTTP
21:33:54.823112 IP 136.178.37.255.45924 > 192.168.1.159.80: Flags [S], seq 631221867:631221987, win 64, length 120: HTTP
21:33:54.823976 IP 110.27.225.146.46013 > 192.168.1.159.80: Flags [S], seq 1733588509:1733588629, win 64, length 120: HTTP
21:33:54.824006 IP 129.208.149.65.46074 > 192.168.1.159.80: Flags [S], seq 1918886471:1918886591, win 64, length 120: HTTP
21:33:54.824327 IP 137.22.133.9.46023 > 192.168.1.159.80: Flags [S], seq 315108302:315108422, win 64, length 120: HTTP
21:33:54.824407 IP 171.220.208.112.46024 > 192.168.1.159.80: Flags [S], seq 754679457:754679577, win 64, length 120: HTTP
21:33:54.824408 IP 171.186.185.119.46025 > 192.168.1.159.80: Flags [S], seq 1894817729:1894817849, win 64, length 120: HTTP
21:33:54.824415 IP 95.136.151.181.46129 > 192.168.1.159.80: Flags [S], seq 1862443648:1862443768, win 64, length 120: HTTP
21:33:54.824447 IP 131.136.211.136.46026 > 192.168.1.159.80: Flags [S], seq 507042471:507042591, win 64, length 120: HTTP
21:33:54.824449 IP 63.149.112.105.46027 > 192.168.1.159.80: Flags [S], seq 568158901:568159021, win 64, length 120: HTTP
21:33:54.824584 IP 158.124.99.26.46032 > 192.168.1.159.80: Flags [S], seq 2025624639:2025624759, win 64, length 120: HTTP
21:33:54.824847 IP 157.128.172.93.46040 > 192.168.1.159.80: Flags [S], seq 1143404615:1143404735, win 64, length 120: HTTP
21:33:54.824880 IP 198.255.146.15.46041 > 192.168.1.159.80: Flags [S], seq 288095268:288095388, win 64, length 120: HTTP
21:33:54.824895 IP 231.200.95.220.46042 > 192.168.1.159.80: Flags [S], seq 1803924015:1803924135, win 64, length 120: HTTP
21:33:54.854742 IP 47.151.81.215.46236 > 192.168.1.159.80: Flags [S], seq 1956165829:1956165949, win 64, length 120: HTTP
21:33:54.872997 IP 146.48.129.158.46361 > 192.168.1.159.80: Flags [S], seq 1995973211:1995973331, win 64, length 120: HTTP
21:33:54.884262 IP 172.52.71.227.46238 > 192.168.1.159.80: Flags [S], seq 1614121324:1614121444, win 64, length 120: HTTP
21:33:54.886123 IP 9.181.38.184.46427 > 192.168.1.159.80: Flags [S], seq 1339339823:1339339943, win 64, length 120: HTTP
21:33:54.891988 IP 48.181.9.3.46239 > 192.168.1.159.80: Flags [S], seq 2021332193:2021332313, win 64, length 120: HTTP
21:33:54.895828 IP 158.120.223.135.46241 > 192.168.1.159.80: Flags [S], seq 315642045:315642165, win 64, length 120: HTTP
21:33:54.900296 IP 231.105.71.228.46242 > 192.168.1.159.80: Flags [S], seq 695710013:695710133, win 64, length 120: HTTP
21:33:54.916739 IP 96.160.96.228.46243 > 192.168.1.159.80: Flags [S], seq 1747331316:1747331436, win 64, length 120: HTTP
21:33:54.919480 IP 217.149.65.164.46245 > 192.168.1.159.80: Flags [S], seq 1733448392:1733448512, win 64, length 120: HTTP
21:33:54.941057 IP 106.91.58.99.46247 > 192.168.1.159.80: Flags [S], seq 729312171:729313391, win 64, length 120: HTTP
21:33:54.943990 IP 48.201.21.228.46249 > 192.168.1.159.80: Flags [S], seq 933095583:933095703, win 64, length 120: HTTP
21:33:54.945114 IP 47.181.213.55.46250 > 192.168.1.159.80: Flags [S], seq 1671217815:1671217935, win 64, length 120: HTTP
21:33:54.951364 IP 149.0.239.118.46277 > 192.168.1.159.80: Flags [S], seq 203158521:203158641, win 64, length 120: HTTP
21:33:54.958697 IP 127.79.12.129.46251 > 192.168.1.159.80: Flags [S], seq 722745875:722745995, win 64, length 120: HTTP
21:33:54.969405 IP 114.117.159.136.46359 > 192.168.1.159.80: Flags [S], seq 1864447032:1864447152, win 64, length 120: HTTP
21:33:54.980786 IP 172.164.227.32.46369 > 192.168.1.159.80: Flags [S], seq 98542542:98542662, win 64, length 120: HTTP
21:33:54.994360 IP 114.124.60.189.46352 > 192.168.1.159.80: Flags [S], seq 1035223587:1035223707, win 64, length 120: HTTP
21:33:55.006566 IP 65.13.111.224.46398 > 192.168.1.159.80: Flags [S], seq 76909556:76909676, win 64, length 120: HTTP
21:33:55.017702 IP 195.149.140.48.46255 > 192.168.1.159.80: Flags [S], seq 166881089:166881129, win 64, length 120: HTTP
21:33:55.037174 IP 225.92.6.112.46258 > 192.168.1.159.80: Flags [S], seq 1493974630:1493974750, win 64, length 120: HTTP
21:33:55.048803 IP 64.151.184.180.46259 > 192.168.1.159.80: Flags [S], seq 1352651590:1352651710, win 64, length 120: HTTP
21:33:55.053102 IP 137.1.159.124.46321 > 192.168.1.159.80: Flags [S], seq 1937290715:1937290835, win 64, length 120: HTTP
21:33:55.087973 IP 58.47.151.95.46260 > 192.168.1.159.80: Flags [S], seq 1772187200:1772187320, win 64, length 120: HTTP
21:33:55.094131 IP 227.54.162.27.46261 > 192.168.1.159.80: Flags [S], seq 1732465732:1732465852, win 64, length 120: HTTP
21:33:55.097000 IP 173.160.13.231.46262 > 192.168.1.159.80: Flags [S], seq 572214795:572214915, win 64, length 120: HTTP
21:33:55.097980 IP 49.184.148.157.46263 > 192.168.1.159.80: Flags [S], seq 1074186212:1074186332, win 64, length 120: HTTP
21:33:55.130569 IP 9.64.250.36.46264 > 192.168.1.159.80: Flags [S], seq 1892567292:1892567412, win 64, length 120: HTTP
21:33:55.144575 IP 65.149.31.148.46265 > 192.168.1.159.80: Flags [S], seq 1551720930:1551721050, win 64, length 120: HTTP
21:33:55.173864 IP 95.248.106.19.46266 > 192.168.1.159.80: Flags [S], seq 438785116:438785236, win 64, length 120: HTTP
21:33:55.215223 IP 193.165.115.227.46267 > 192.168.1.159.80: Flags [S], seq 509925679:509925799, win 64, length 120: HTTP
21:33:55.224922 IP 64.49.249.157.46268 > 192.168.1.159.80: Flags [S], seq 1548724422:1548724542, win 64, length 120: HTTP
21:33:55.232048 IP 52.45.221.214.46270 > 192.168.1.159.80: Flags [S], seq 508090725:508090845, win 64, length 120: HTTP
```

```
File  Edit  View  Search  Terminal  Help
21:35:28.456438 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 24702, length 8
21:35:28.456670 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 24958, length 8
21:35:28.456690 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 25214, length 8
21:35:28.456700 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 25470, length 8
21:35:28.456740 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 25726, length 8
21:35:28.456795 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 25982, length 8
21:35:28.456815 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 26238, length 8
21:35:28.456851 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 26494, length 8
21:35:28.463890 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 62590, length 8
21:35:28.464617 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 62846, length 8
21:35:28.465580 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 63102, length 8
21:35:28.466486 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 63358, length 8
21:35:28.467213 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 63614, length 8
21:35:28.467922 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 63870, length 8
21:35:28.468968 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 64126, length 8
21:35:28.469788 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 64382, length 8
21:35:28.470577 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 64638, length 8
21:35:28.471604 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 64894, length 8
21:35:28.472663 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 65150, length 8
21:35:28.473435 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 65406, length 8
21:35:28.474306 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 127, length 8
21:35:28.475142 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 383, length 8
21:35:28.475952 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 639, length 8
21:35:28.476936 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 895, length 8
21:35:28.477922 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 1151, length 8
21:35:28.478762 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 1407, length 8
21:35:28.479781 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 1663, length 8
21:35:28.481145 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 1919, length 8
21:35:28.482659 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 2175, length 8
21:35:28.484017 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 2431, length 8
21:35:28.486223 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 2687, length 8
21:35:28.488084 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 2943, length 8
21:35:28.493669 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 3199, length 8
21:35:28.495720 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 3455, length 8
21:35:28.495722 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 3711, length 8
21:35:28.495723 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 3967, length 8
21:35:28.495724 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 4223, length 8
21:35:28.495725 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 4479, length 8
21:35:28.495726 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 4735, length 8
21:35:28.504753 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 8063, length 8
21:35:28.504768 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 8319, length 8
21:35:28.504769 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 8575, length 8
21:35:28.504771 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 8831, length 8
21:35:28.504772 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 9087, length 8
21:35:28.504773 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 9343, length 8
21:35:28.504774 IP 192.168.103.1 > 192.168.1.255: ICMP echo request, id 45065, seq 9599, length 8
```

Conclusion:-Learnt more about the network analysis and security assessment tools. Explored various network probing and testing techniques, which are valuable skills in the field of network administration and cybersecurity.Also executed several hping3 commands and performed DOS attack using hping3

Animesh Parab T2-T21- 88

Lab Assignment 10

Aim: To study and configure Firewalls using IP tables

LO Attainment **: LO6**

Firewall:

A firewall is a system designed to prevent unauthorized access to or from a private network. You can implement a firewall in either hardware or software form, or a combination of both. Generally the firewall has two network interfaces: one for the external side of the network, one for the internal side. Its purpose is to control what traffic is allowed to traverse from one side to the other. As the most basic level, firewalls can block traffic intended for particular IP addresses or server ports.

TCP network traffic moves around a network in packets, which are containers that consist of a packet header—this contains control information such as source and destination addresses, and packet sequence information—and the data (also known as a payload). While the control information in each packet helps to ensure that its associated data gets delivered properly, the elements it contains also provides firewalls a variety of ways to match packets against firewall rules.

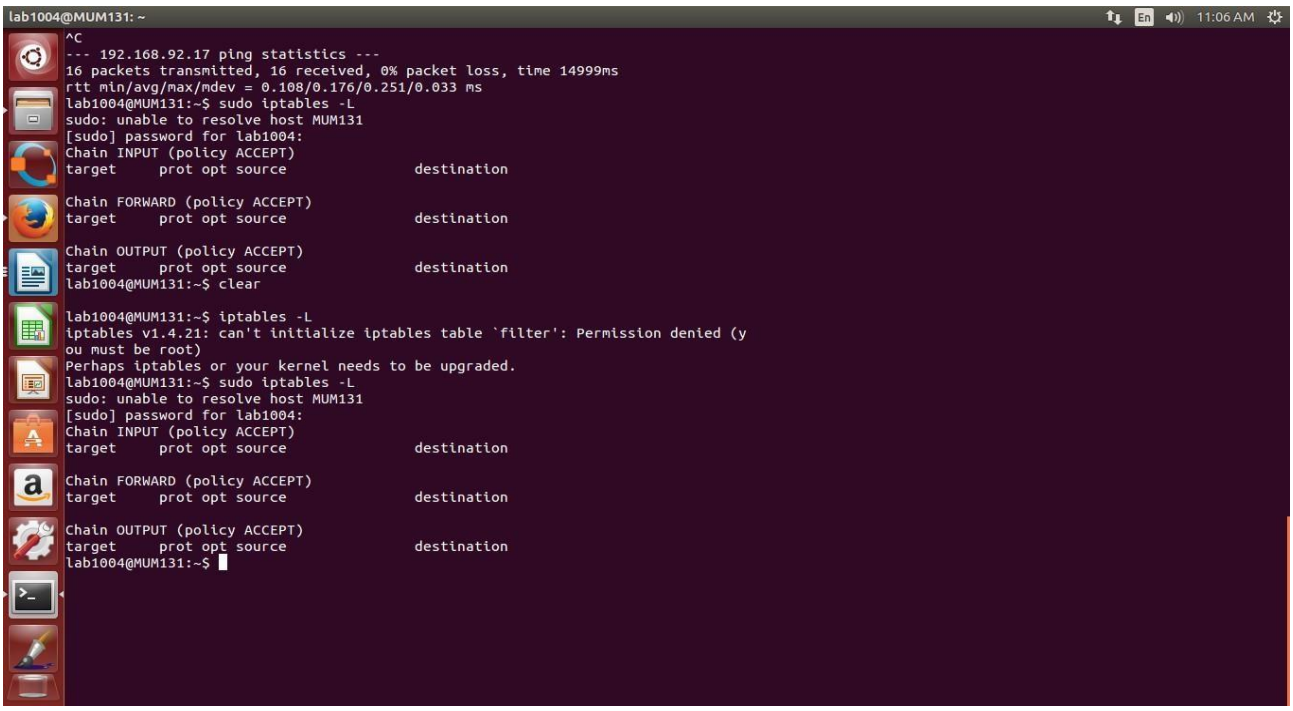
Types of Firewalls

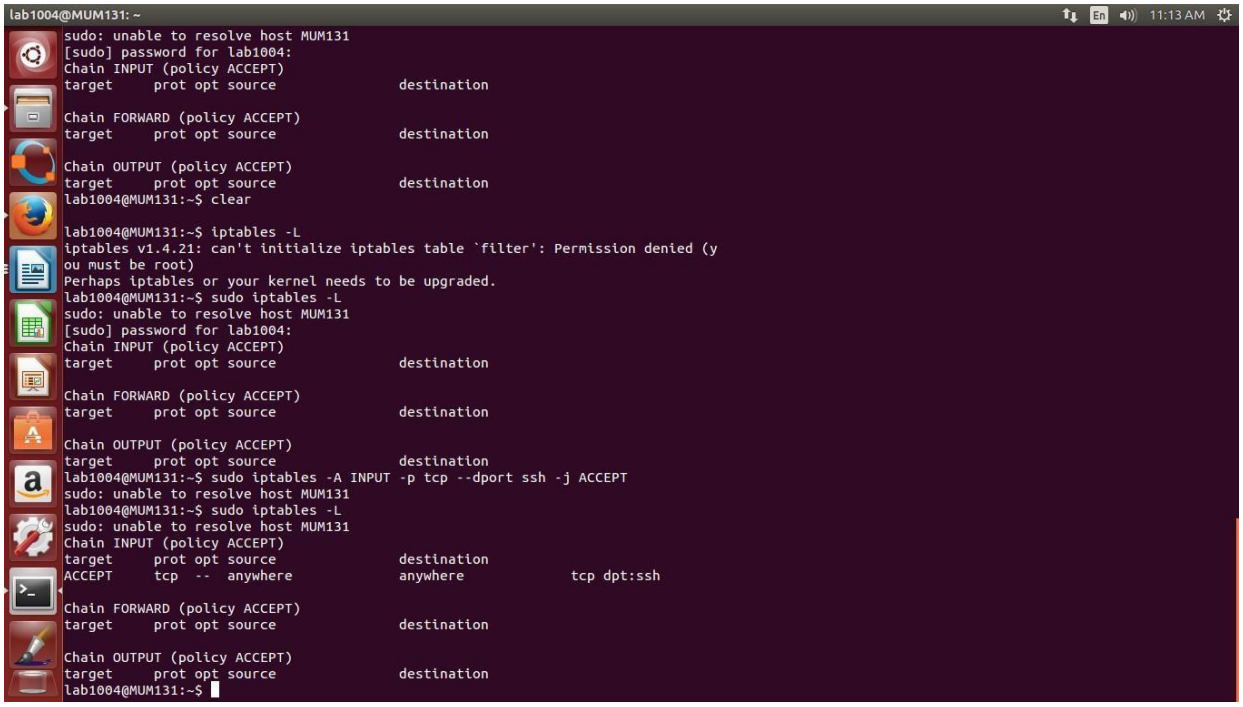
Three basic types of network firewalls: packet filtering (stateless), stateful, and application layer.

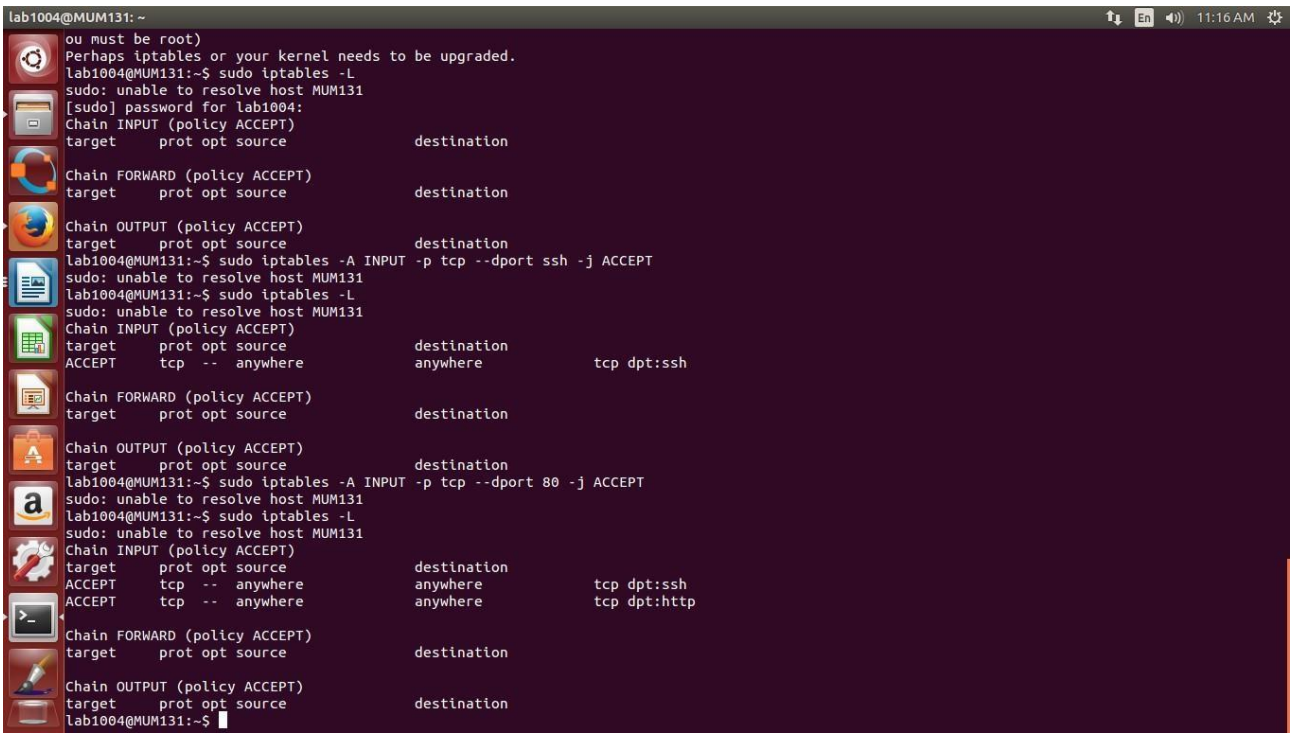
Packet filtering, or stateless, firewalls work by inspecting individual packets in isolation. As such, they are unaware of connection state and can only allow or deny packets based on individual packet headers.

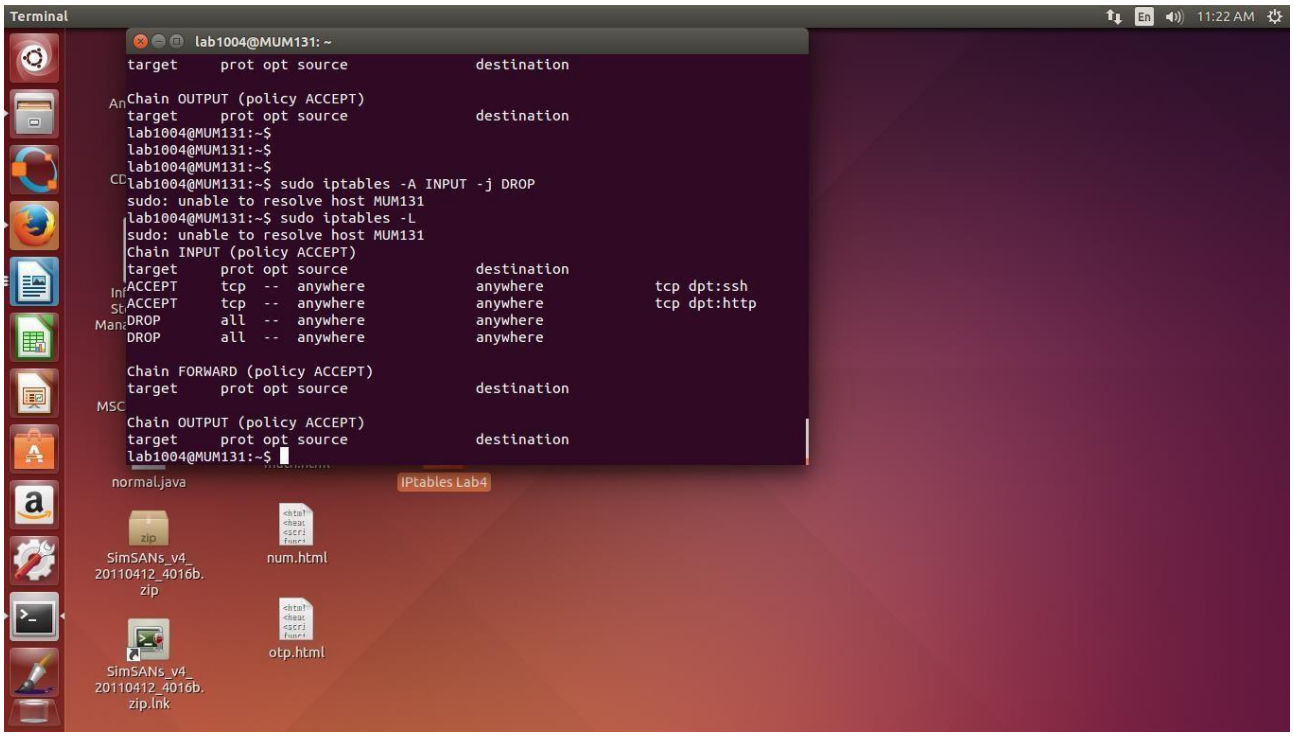
Stateful firewalls are able to determine the connection state of packets, which makes them much more flexible than stateless firewalls. They work by collecting related packets until the connection state can be determined before any firewall rules are applied to the traffic.

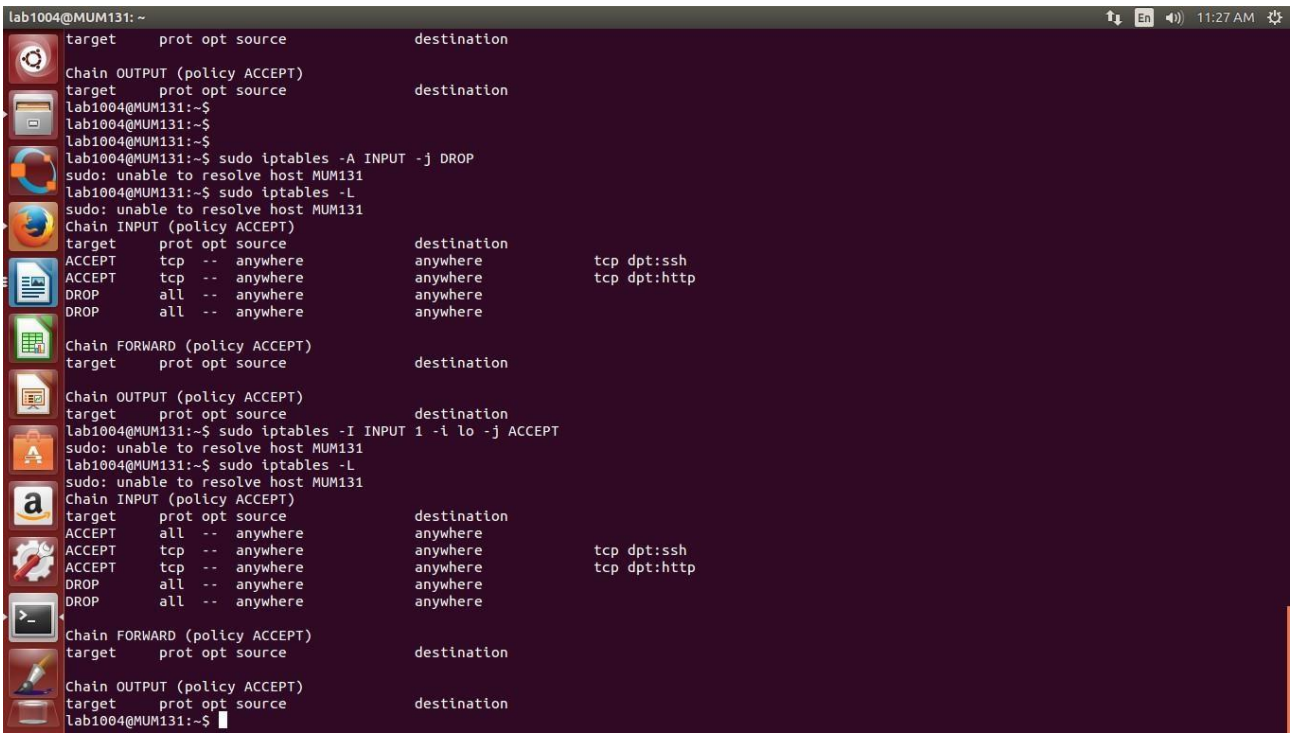
Application firewalls go one step further by analyzing the data being transmitted, which allows network traffic to be matched against firewall rules that are specific to individual services or applications. These are also known as proxy-based firewalls.

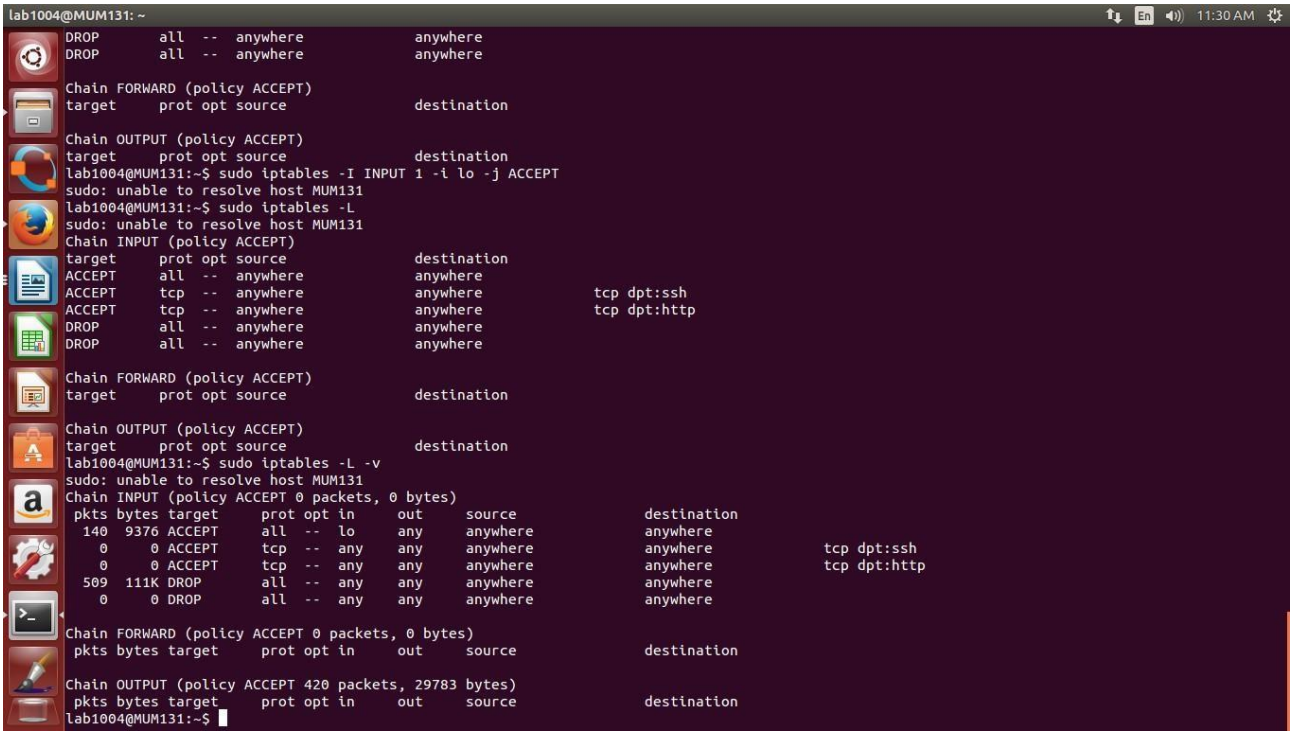

lab1004@MUM131: ~
11:06 AM
^C
--- 192.168.92.17 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 14999ms
rtt min/avg/max/mdev = 0.108/0.176/0.251/0.033 ms
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ clear
lab1004@MUM131:~$ iptables -L
iptables v1.4.21: can't initialize iptables table `filter': Permission denied (y
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$

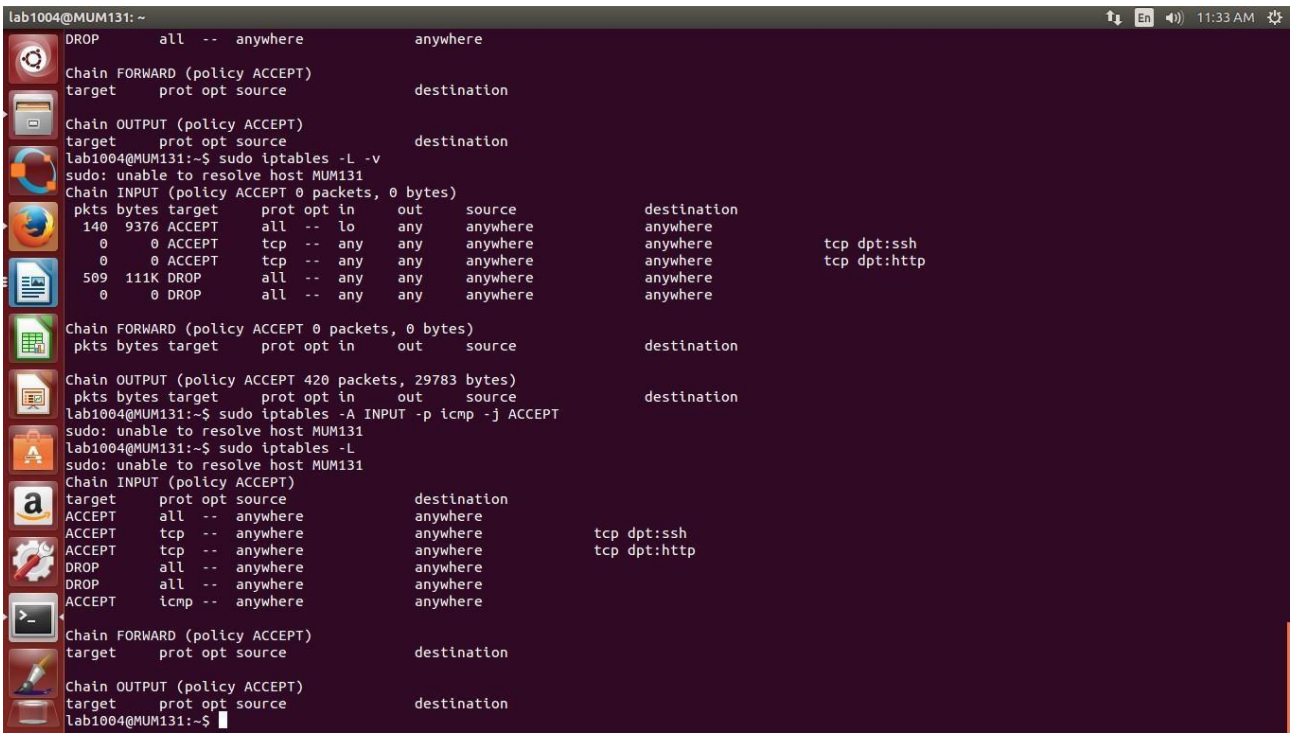

lab1004@MUM131: ~
11:13 AM
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ clear
lab1004@MUM131:~$ iptables -L
iptables v1.4.21: can't initialize iptables table `filter': Permission denied (y
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$

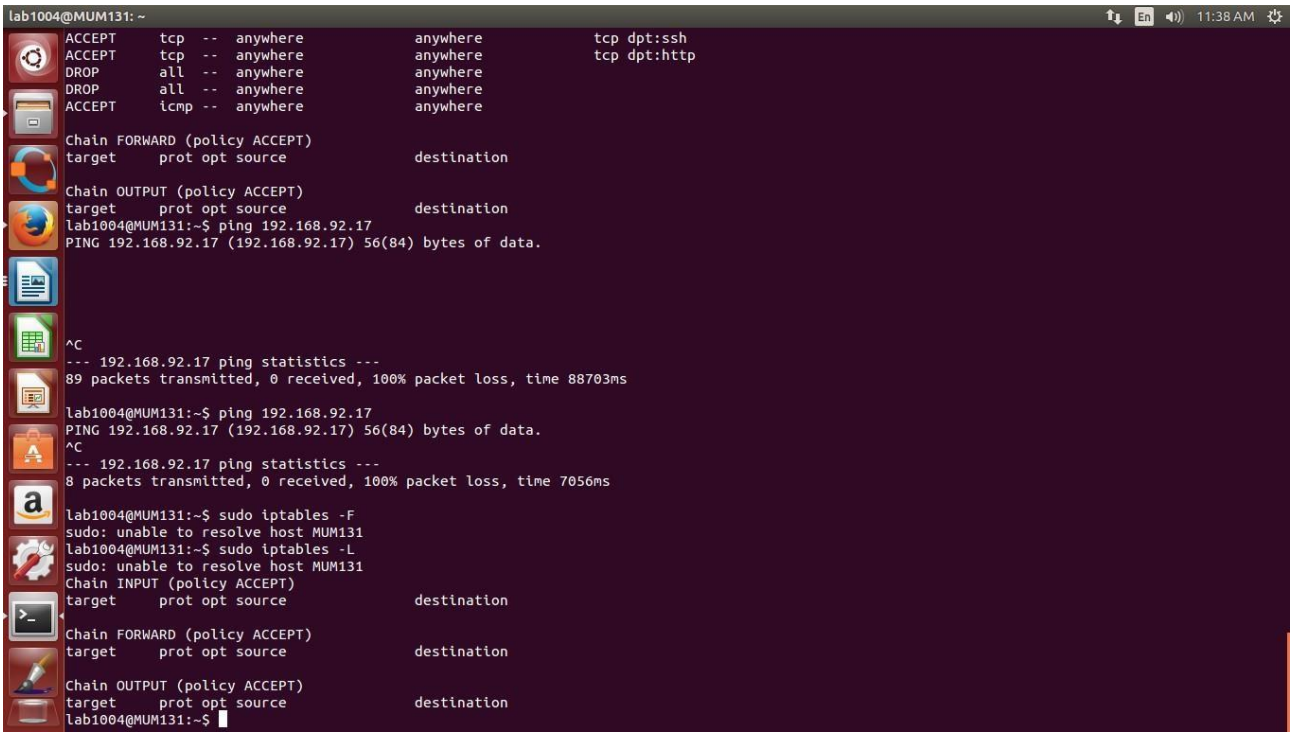

lab1004@MUM131: ~
11:16 AM
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$

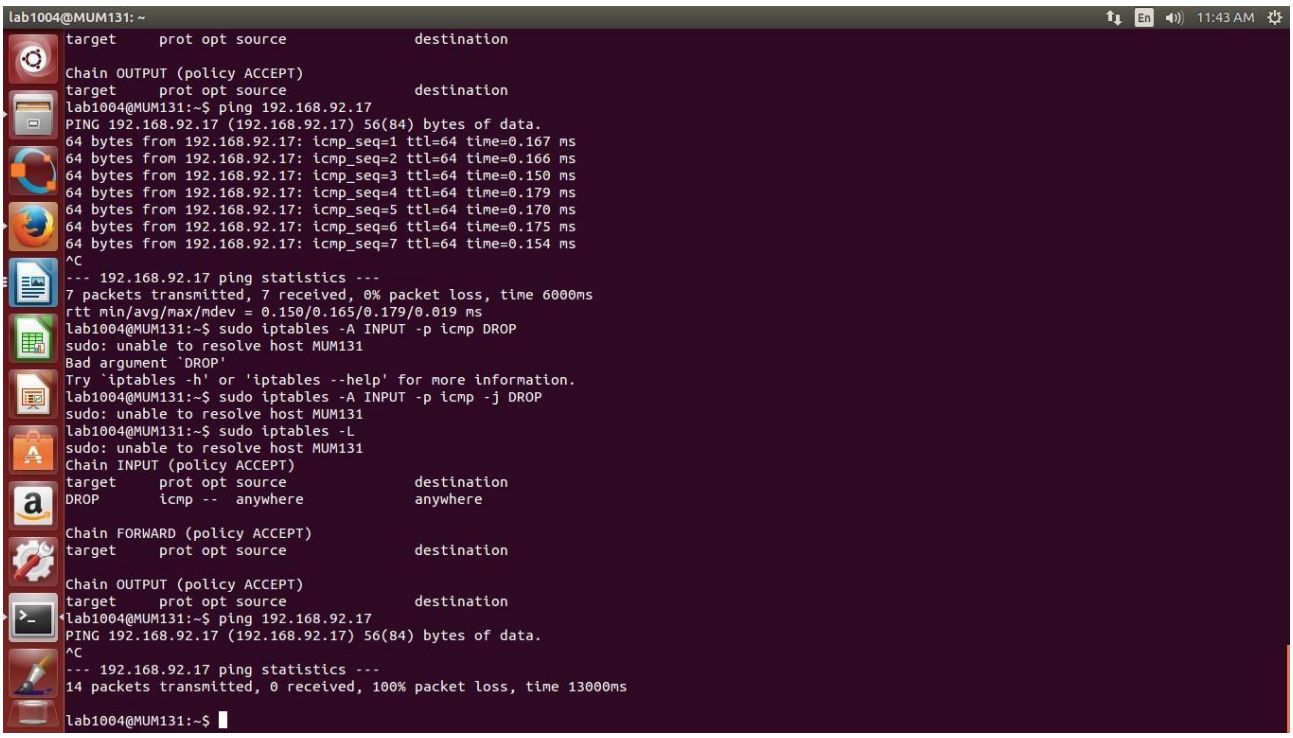

Terminal
11:22 AM
lab1004@MUM131: ~
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$ sudo iptables -A INPUT -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$
normal.java
IPtables Lab4
SimSANs_v4_20110412_4016b.zip
num.html
otp.html
SimSANs_v4_20110412_4016b.zip.lnk

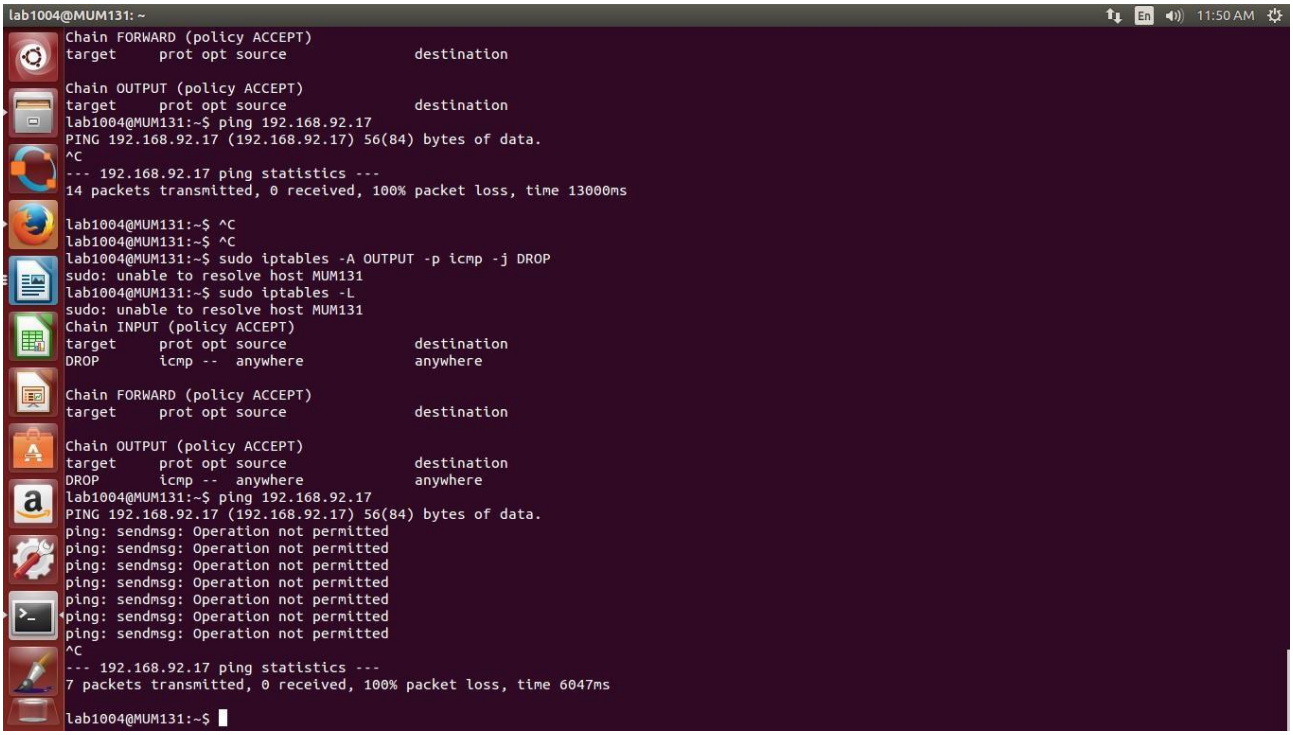

lab1004@MUM131: ~
11:27 AM
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$ sudo iptables -A INPUT -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT all -- anywhere anywhere
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$

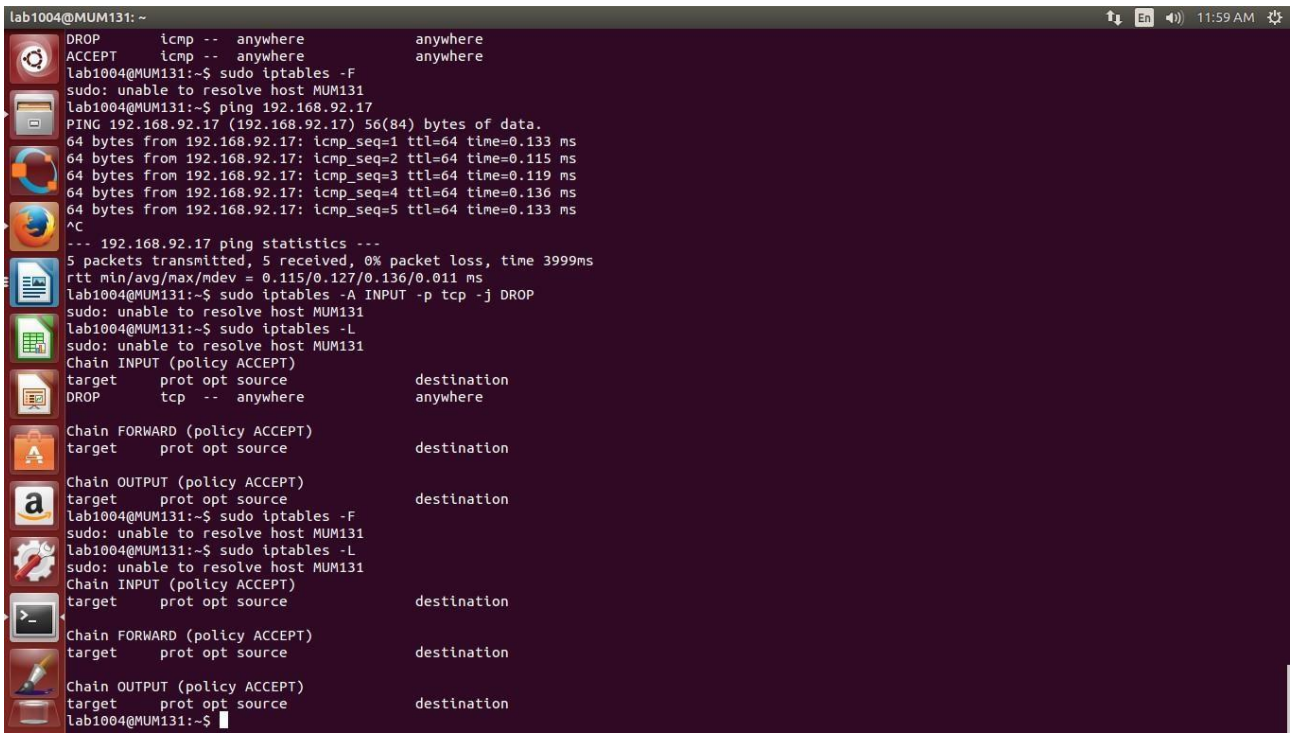

lab1004@MUM131: ~
11:30 AM
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT all -- anywhere anywhere
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -L -v
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
140 9376 ACCEPT all -- lo any anywhere anywhere
0 0 ACCEPT tcp -- any any anywhere anywhere tcp dpt:ssh
0 0 ACCEPT tcp -- any any anywhere anywhere tcp dpt:http
509 111K DROP all -- any any anywhere anywhere
0 0 DROP all -- any any anywhere anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
Chain OUTPUT (policy ACCEPT 420 packets, 29783 bytes)
pkts bytes target prot opt in out source destination
lab1004@MUM131:~$


lab1004@MUM131: ~
11:33 AM
DROP all -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ sudo iptables -L -v
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
140 9376 ACCEPT all -- lo any anywhere anywhere
0 0 ACCEPT tcp -- any any anywhere anywhere tcp dpt:ssh
0 0 ACCEPT tcp -- any any anywhere anywhere tcp dpt:http
509 111K DROP all -- any any anywhere anywhere
0 0 DROP all -- any any anywhere anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
Chain OUTPUT (policy ACCEPT 420 packets, 29783 bytes)
pkts bytes target prot opt in out source destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
ACCEPT all -- anywhere anywhere
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
ACCEPT icmp -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$


lab1004@MUM131: ~
11:38 AM
ACCEPT tcp -- anywhere anywhere tcp dpt:ssh
ACCEPT tcp -- anywhere anywhere tcp dpt:http
DROP all -- anywhere anywhere
DROP all -- anywhere anywhere
ACCEPT icmp -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
89 packets transmitted, 0 received, 100% packet loss, time 88703ms
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7056ms
lab1004@MUM131:~$ sudo iptables -F
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$


lab1004@MUM131: ~
11:43 AM
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
64 bytes from 192.168.92.17: icmp_seq=1 ttl=64 time=0.167 ms
64 bytes from 192.168.92.17: icmp_seq=2 ttl=64 time=0.166 ms
64 bytes from 192.168.92.17: icmp_seq=3 ttl=64 time=0.150 ms
64 bytes from 192.168.92.17: icmp_seq=4 ttl=64 time=0.179 ms
64 bytes from 192.168.92.17: icmp_seq=5 ttl=64 time=0.170 ms
64 bytes from 192.168.92.17: icmp_seq=6 ttl=64 time=0.175 ms
64 bytes from 192.168.92.17: icmp_seq=7 ttl=64 time=0.154 ms
^C
--- 192.168.92.17 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6000ms
rtt min/avg/max/mdev = 0.150/0.165/0.179/0.019 ms
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp DROP
sudo: unable to resolve host MUM131
Bad argument `DROP'
Try `iptables -h' or 'iptables --help' for more information.
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
DROP icmp -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
14 packets transmitted, 0 received, 100% packet loss, time 13000ms
lab1004@MUM131:~$


lab1004@MUM131: ~
11:50 AM
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
14 packets transmitted, 0 received, 100% packet loss, time 13000ms
lab1004@MUM131:~$ ^C
lab1004@MUM131:~$ ^C
lab1004@MUM131:~$ sudo iptables -A OUTPUT -p icmp -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target prot opt source destination
DROP icmp -- anywhere anywhere
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
DROP icmp -- anywhere anywhere
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 192.168.92.17 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6047ms
lab1004@MUM131:~$



Conclusion :-

We have successfully learned and implemented the concept of firewalls, we learned to see content of iptables ,get more details of the table, append new rules for packet filteration, droping and blocking the packets of specific protocol, etc.

Animesh Parab T2-T21 88

LAB ASSIGNMENT No. 11

Aim: Installing snort, configuring it in Intrusion Detection mode and writing rules for detecting pinging activity.

Lab Outcome Attained: LO6 **Theory:**

Steps to Install snort and configure it in Intrusion Detection Mode.

1. Check the name of the interface using command ifconfig.
2. Install snort in ubuntu machine using command *sudo apt-get install snort*
3. While installing the snort, name of the interface will be asked on which snort is supposed to listen. Enter the interface name observed in step 1.
4. Run the command *sudo gedit /etc/snort/snort.conf* . This opens snort configuration file.
5. Make following changes to configuration file.

   a. ipvar HOME_NET **192.168.0.0/24 (in section 1)**
6. Open new terminal. Open ftp.rule file in it by typing the command *sudo gedit /etc/snort/rules/ftp.rules (optional)*
7. Open new terminal and type the command *sudo snort -T -c /etc/snort/snort.conf -i* ***enp3s0*** to validate that all rules are there.

   We use the

   -T flag to test the configuration file,

   -c flag to tell Snort which configuration file to use, and -i to specify the interface that Snort will listen on.

8. Type the command *sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp3s0* (to start snort in NIDS mode)

   We use the

   -A console The 'console' option prints fast mode alerts to stdout -q Quiet mode. Don't show banner and status report.

   -u snort Run Snort as the following user after startup

-g snort Run Snort as the following group after startup

-c /etc/snort/snort.conf The path to our snort.conf file

-i enp3s0 The interface to listen on (change to your interface if different)

9. Now go to kali linux machine.

10. Type command *nmap 192.168.0.107* on it to start port scanning of ubuntu machine and observe the output in terminal where snort is started in detection environment.

When you execute this command, you will not initially see any output. Snort is running, and is processing all packets that arrive on eth0 (or whichever interface you specified with the -i flag). Snort compares each packet to the rules it has loaded (in this case our single ICMP Ping rule), and will then print an alert to the console when a packet matches our rule.

11. Then try pinging ubuntu machine by typing the command *ping 192.168.0.107* and observe the output in terminal where snort is started in detection mode.

---

12. Adding rule for detecting ping activity performed by another machine:

---

a. In ubuntu machine, type the following command to create a file called local.rules : ***sudo gedit /etc/snort/rules/local.rules***

b. Write the following rule in it: ***alert icmp any any -> $HOME_NET any (msg:"ICMP test detected"; GID:1; sid:10000001; rev:001; classtype:icmp-event;)***

c. Save the local.rules file.

d. Comment the following lines in configuration file (snort.conf) of snort: icmp.rules and icmp-info.rules

e. Add the local.rules file in section 7 of configuration file of snort by writing: ***include $RULE_PATH local.rules***

f. Validate the changes made in snort.conf file by writing the command in terminal: ***sudo snort -T -c /etc/snort/snort.conf -i enp3s0***

g. Set the snort in Intrusion Detection Mode by typing the command: ***sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf i enp3s0***

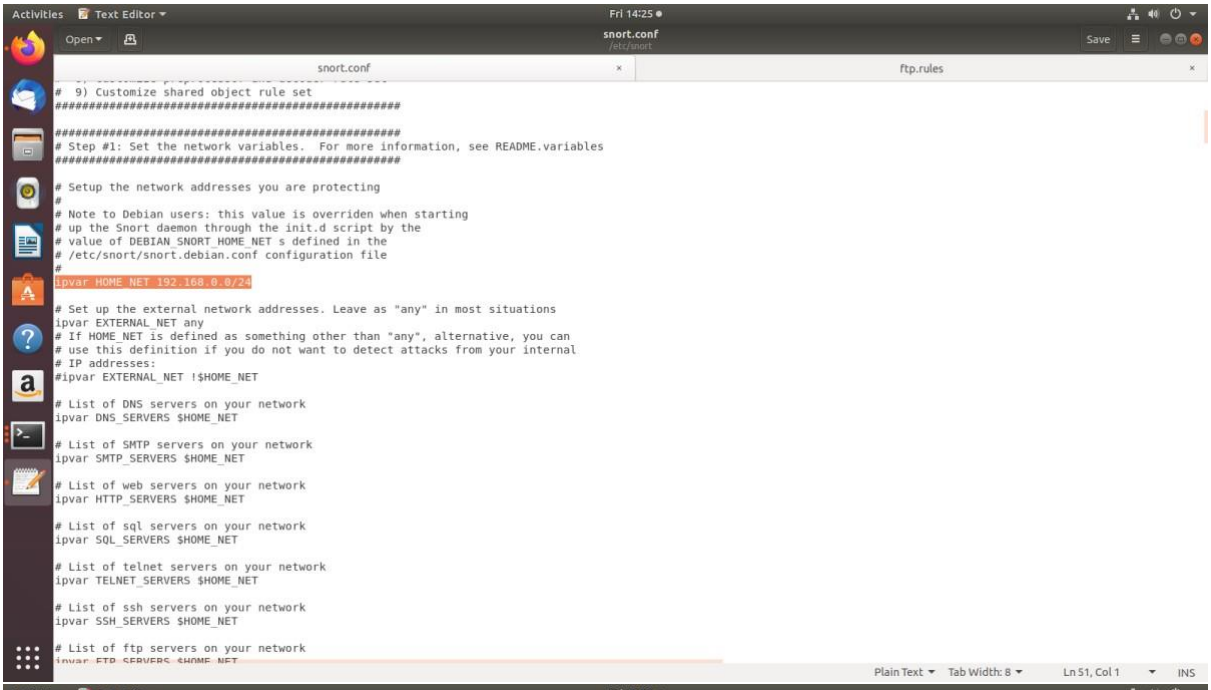
h. Now from kali machine ping the ubuntu machine and see the alert generated.

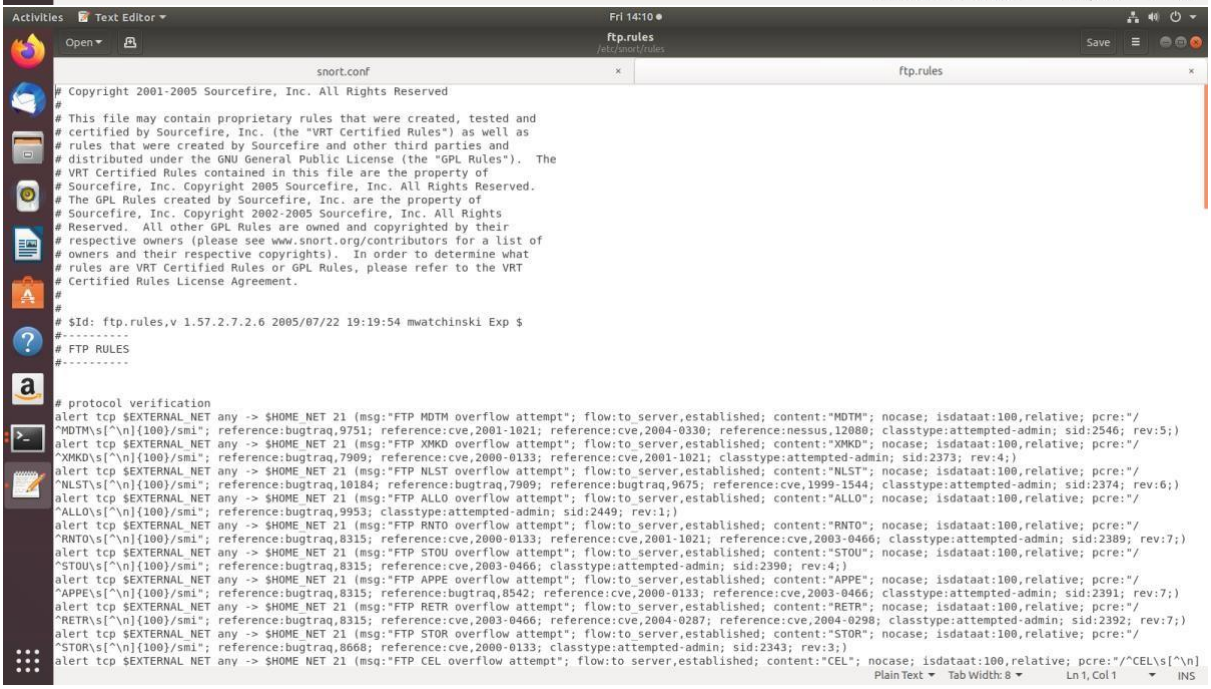
i. Observe the difference between the alerts generated when icmp.rules and icmp-info.rules are used and when local.rules is used to detect the ping activity.

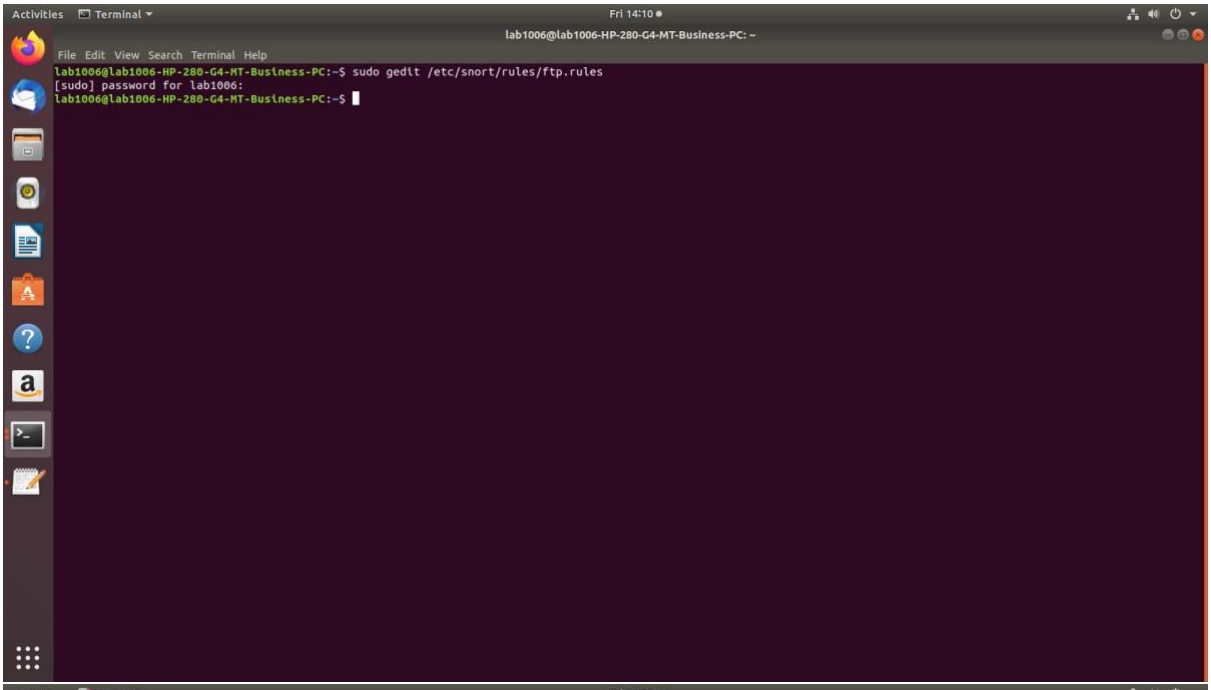
Reference Link for Demo: https://www.youtube.com/watch?v=iBsGSsbDMyw

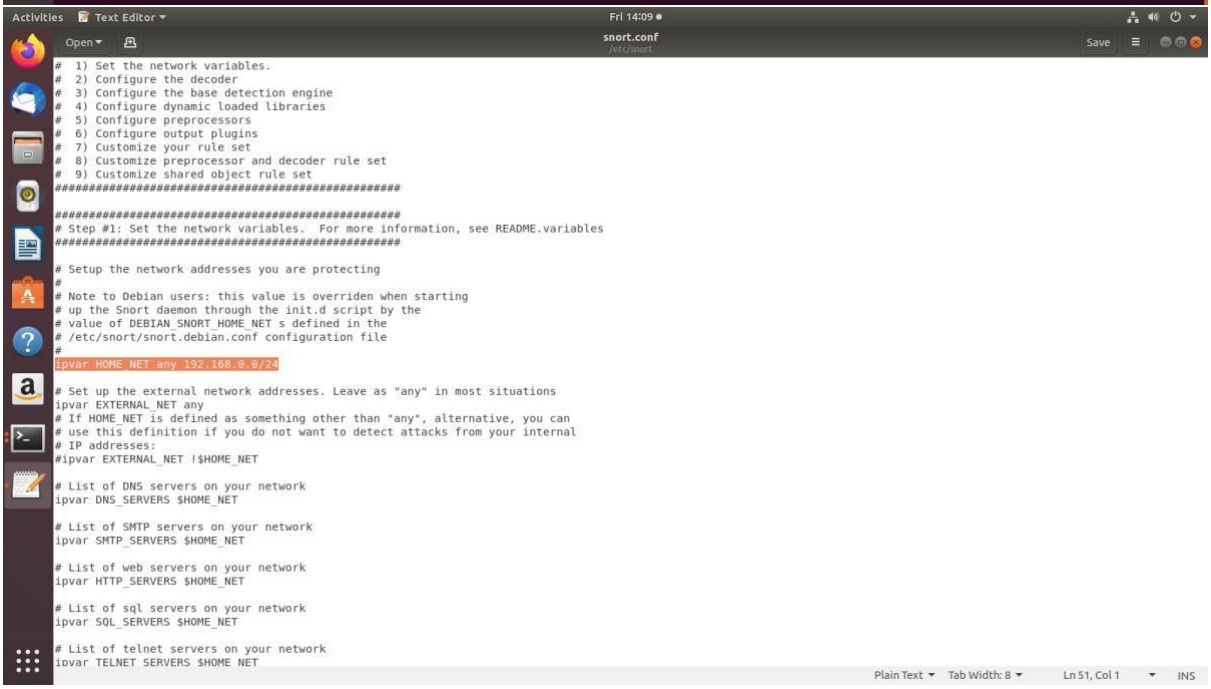
---

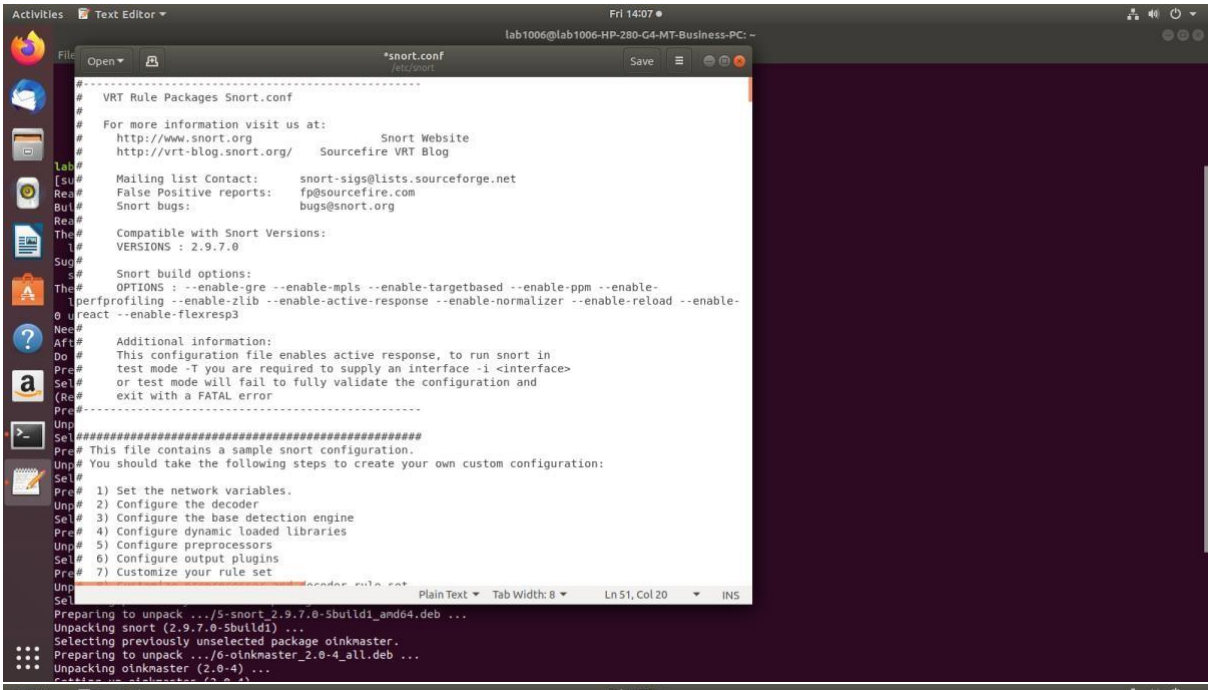
Output:

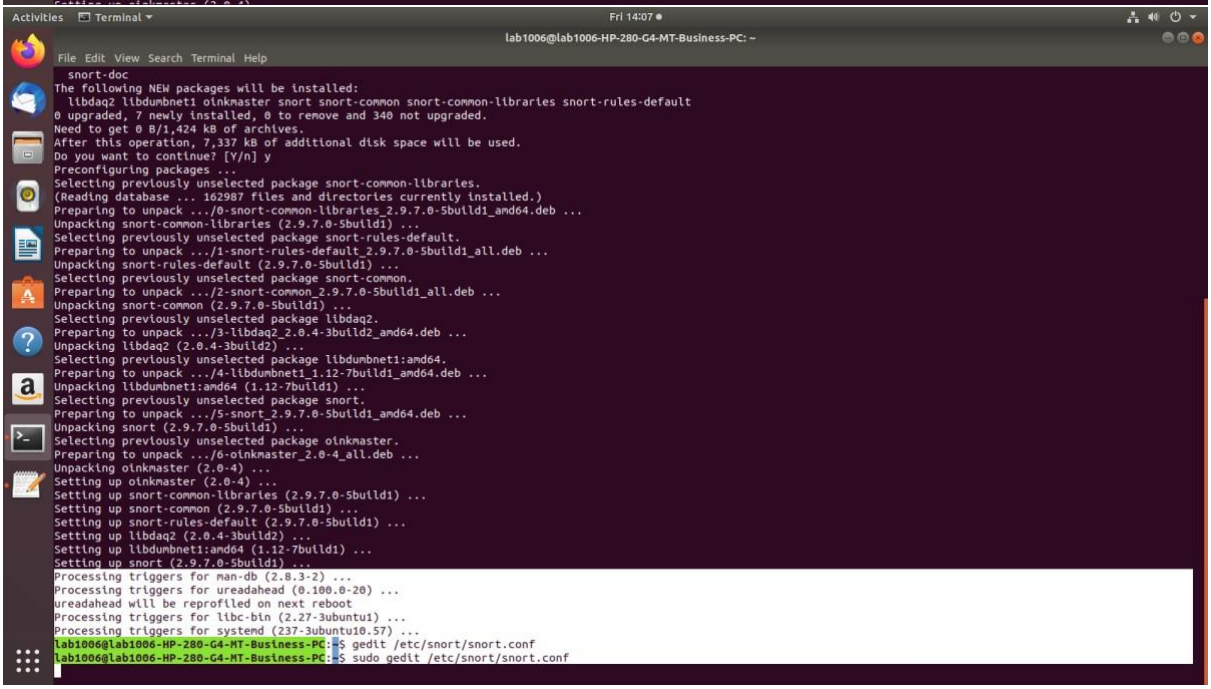

Activities
Text Editor
Fri 14:25
Open
snort.conf
/etc/snort
Save
snort.conf
ftp.rules
9) Customize shared object rule set
##
##
Step #1: Set the network variables. For more information, see README.variables
##
Setup the network addresses you are protecting
#
Note to Debian users: this value is overriden when starting
up the Snort daemon through the init.d script by the
value of DEBIAN_SNORT_HOME_NET s defined in the
/etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.0.0/24
Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
If HOME_NET is defined as something other than "any", alternative, you can
use this definition if you do not want to detect attacks from your internal
IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET
List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET
List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET
List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET
List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET
List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET
List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET
List of ftp servers on your network
Plain Text
Tab Width: 8
Ln 51, Col 1
INS

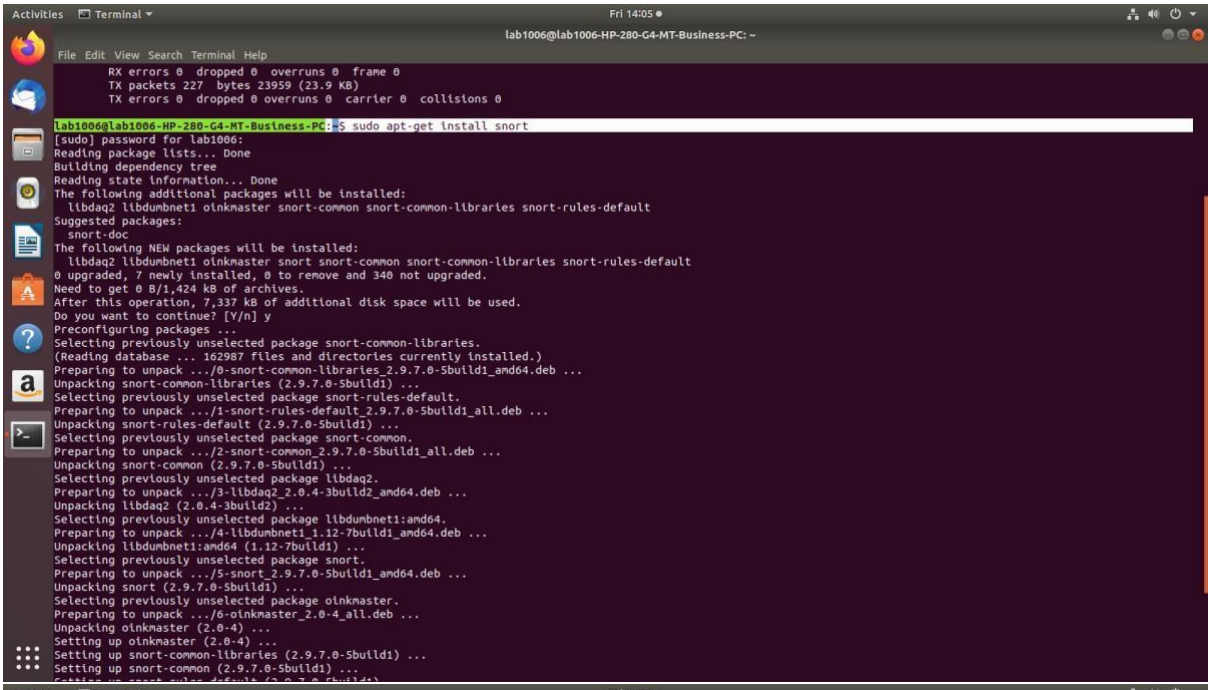

Activities
Text Editor
Fri 14:10
Open
ftp.rules
/etc/snort/rules
Save
snort.conf
ftp.rules
Copyright 2001-2005 Sourcefire, Inc. All Rights Reserved
#
This file may contain proprietary rules that were created, tested and
certified by Sourcefire, Inc. (the "VRT Certified Rules") as well as
rules that were created by Sourcefire and other third parties and
distributed under the GNU General Public License (the "GPL Rules"). The
VRT Certified Rules contained in this file are the property of
Sourcefire, Inc. Copyright 2005 Sourcefire, Inc. All Rights Reserved.
The GPL Rules created by Sourcefire, Inc. are the property of
Sourcefire, Inc. Copyright 2002-2005 Sourcefire, Inc. All Rights
Reserved. All other GPL Rules are owned and copyrighted by their
respective owners (please see www.snort.org/contributors for a list of
owners and their respective copyrights). In order to determine what
rules are VRT Certified Rules or GPL Rules, please refer to the VRT
Certified Rules License Agreement.
#
#
$Id: ftp.rules,v 1.57.2.7.2.6 2005/07/22 19:19:54 mwatchinski Exp $
#----------
FTP RULES
#----------
protocol verification
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP MDTM overflow attempt"; flow:to_server,established; content:"MDTM"; nocase; isdataat:100,relative; pcre:"/
^MDTM\s[^\n]{100}/smi"; reference:bugtraq,9751; reference:cve,2001-1021; reference:cve,2004-0330; reference:nessus,12080; classtype:attempted-admin; sid:2546; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP XMKD overflow attempt"; flow:to_server,established; content:"XMKD"; nocase; isdataat:100,relative; pcre:"/
^XMKD\s[^\n]{100}/smi"; reference:bugtraq,7909; reference:cve,2000-0133; reference:cve,2001-1021; classtype:attempted-admin; sid:2373; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP NLST overflow attempt"; flow:to_server,established; content:"NLST"; nocase; isdataat:100,relative; pcre:"/
^NLST\s[^\n]{100}/smi"; reference:bugtraq,10184; reference:bugtraq,7909; reference:bugtraq,9675; reference:cve,1999-1544; classtype:attempted-admin; sid:2374; rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP ALLO overflow attempt"; flow:to_server,established; content:"ALLO"; nocase; isdataat:100,relative; pcre:"/
^ALLO\s[^\n]{100}/smi"; reference:bugtraq,9953; classtype:attempted-admin; sid:2449; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP RNTO overflow attempt"; flow:to_server,established; content:"RNTO"; nocase; isdataat:100,relative; pcre:"/
^RNTO\s[^\n]{100}/smi"; reference:bugtraq,8315; reference:cve,2000-0133; reference:cve,2001-1021; reference:cve,2003-0466; classtype:attempted-admin; sid:2389; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP STOU overflow attempt"; flow:to_server,established; content:"STOU"; nocase; isdataat:100,relative; pcre:"/
^STOU\s[^\n]{100}/smi"; reference:bugtraq,8315; reference:cve,2003-0466; classtype:attempted-admin; sid:2390; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP APPE overflow attempt"; flow:to_server,established; content:"APPE"; nocase; isdataat:100,relative; pcre:"/
^APPE\s[^\n]{100}/smi"; reference:bugtraq,8315; reference:bugtraq,8542; reference:cve,2000-0133; reference:cve,2003-0466; classtype:attempted-admin; sid:2391; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP RETR overflow attempt"; flow:to_server,established; content:"RETR"; nocase; isdataat:100,relative; pcre:"/
^RETR\s[^\n]{100}/smi"; reference:bugtraq,8315; reference:cve,2003-0466; reference:cve,2004-0287; reference:cve,2004-0298; classtype:attempted-admin; sid:2392; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP STOR overflow attempt"; flow:to_server,established; content:"STOR"; nocase; isdataat:100,relative; pcre:"/
^STOR\s[^\n]{100}/smi"; reference:bugtraq,8668; reference:cve,2000-0133; classtype:attempted-admin; sid:2343; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CEL overflow attempt"; flow:to server,established; content:"CEL"; nocase; isdataat:100,relative; pcre:"/^CEL\s[^\n]
Plain Text
Tab Width: 8
Ln 1, Col 1
INS

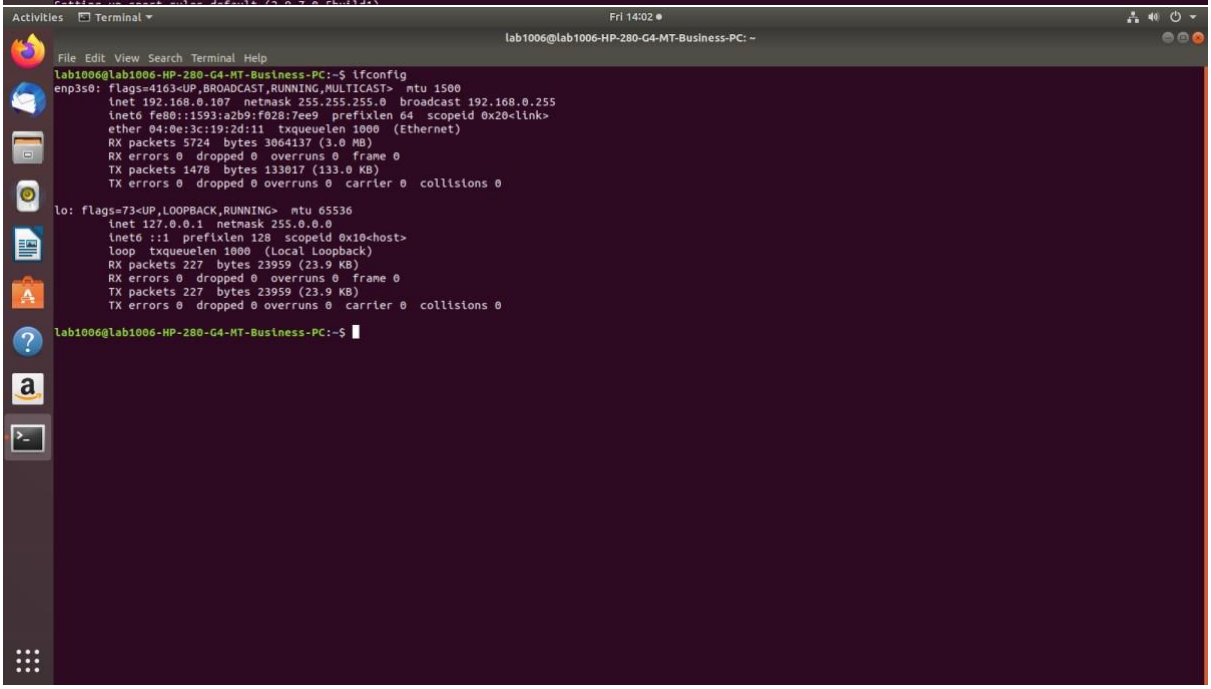

Activities
Terminal
Fri 14:10
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo gedit /etc/snort/rules/ftp.rules
[sudo] password for lab1006:
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$


Activities
Text Editor
Fri 14:09
Open
snort.conf
/etc/snort
Save
1) Set the network variables.
2) Configure the decoder
3) Configure the base detection engine
4) Configure dynamic loaded libraries
5) Configure preprocessors
6) Configure output plugins
7) Customize your rule set
8) Customize preprocessor and decoder rule set
9) Customize shared object rule set
Step #1: Set the network variables. For more information, see README.variables
Setup the network addresses you are protecting
Note to Debian users: this value is overriden when starting
up the Snort daemon through the init.d script by the
value of DEBIAN_SNORT_HOME_NET s defined in the
/etc/snort/snort.debian.conf configuration file
ipvar HOME_NET any 192.168.0.0/24
Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
If HOME_NET is defined as something other than "any", alternative, you can
use this definition if you do not want to detect attacks from your internal
IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET
List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET
List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET
List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET
List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET
List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET
Plain Text
Tab Width: 8
Ln 51, Col 1
INS


Activities
Text Editor
Fri 14:07
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
Open
*snort.conf
/etc/snort
Save
VRT Rule Packages Snort.conf
For more information visit us at:
http://www.snort.org Snort Website
http://vrt-blog.snort.org/ Sourcefire VRT Blog
Mailing list Contact: snort-sigs@lists.sourceforge.net
False Positive reports: fp@sourcefire.com
Snort bugs: bugs@snort.org
Compatible with Snort Versions:
VERSIONS : 2.9.7.0
Snort build options:
OPTIONS : --enable-gre --enable-mpls --enable-targetbased --enable-ppm --enable-perfprofiling --enable-zlib --enable-active-response --enable-normalizer --enable-reload --enable-react --enable-flexresp3
Additional information:
This configuration file enables active response, to run snort in
test mode -T you are required to supply an interface -i <interface>
or test mode will fail to fully validate the configuration and
exit with a FATAL error
This file contains a sample snort configuration.
You should take the following steps to create your own custom configuration:
1) Set the network variables.
2) Configure the decoder
3) Configure the base detection engine
4) Configure dynamic loaded libraries
5) Configure preprocessors
6) Configure output plugins
7) Customize your rule set
Plain Text
Tab Width: 8
Ln 51, Col 20
INS
Preparing to unpack .../5-snort_2.9.7.0-5build1_amd64.deb ...
Unpacking snort (2.9.7.0-5build1) ...
Selecting previously unselected package oinkmaster.
Preparing to unpack .../6-oinkmaster_2.0-4_all.deb ...
Unpacking oinkmaster (2.0-4) ...

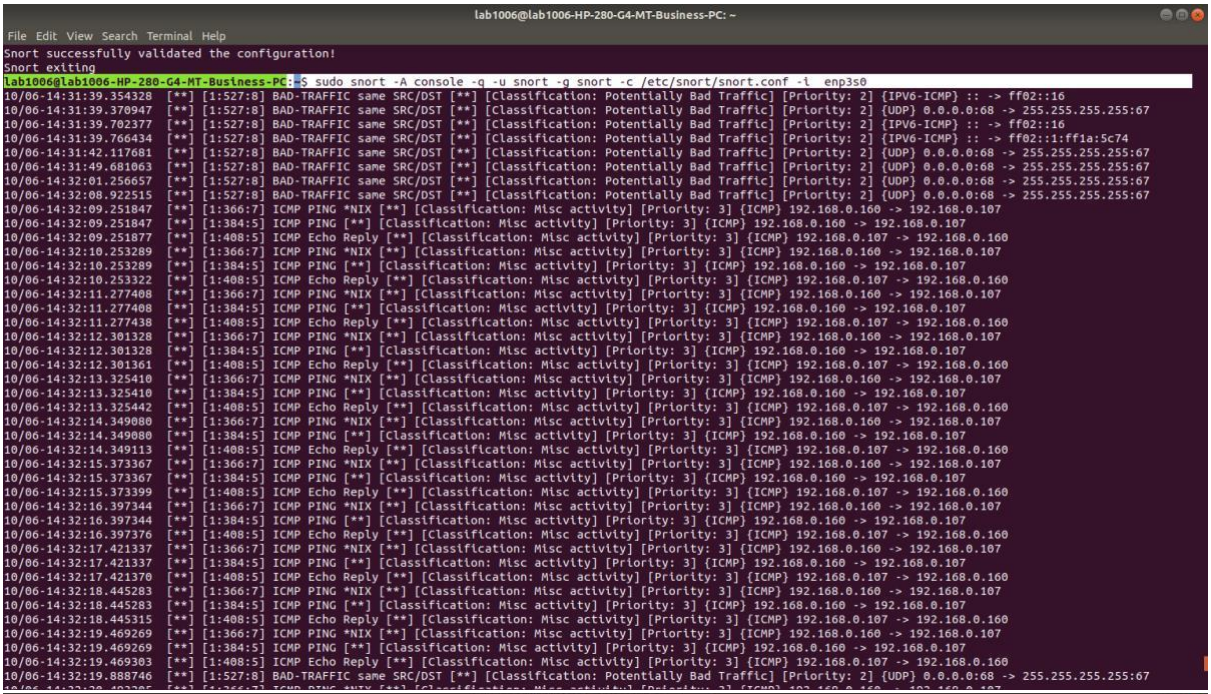

Activities
Terminal
Fri 14:07
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
snort-doc
The following NEW packages will be installed:
libdaq2 libdumbnet1 oinkmaster snort snort-common snort-common-libraries snort-rules-default
0 upgraded, 7 newly installed, 0 to remove and 340 not upgraded.
Need to get 0 B/1,424 kB of archives.
After this operation, 7,337 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Preconfiguring packages ...
Selecting previously unselected package snort-common-libraries.
(Reading database ... 162987 files and directories currently installed.)
Preparing to unpack .../0-snort-common-libraries_2.9.7.0-5build1_amd64.deb ...
Unpacking snort-common-libraries (2.9.7.0-5build1) ...
Selecting previously unselected package snort-rules-default.
Preparing to unpack .../1-snort-rules-default_2.9.7.0-5build1_all.deb ...
Unpacking snort-rules-default (2.9.7.0-5build1) ...
Selecting previously unselected package snort-common.
Preparing to unpack .../2-snort-common_2.9.7.0-5build1_all.deb ...
Unpacking snort-common (2.9.7.0-5build1) ...
Selecting previously unselected package libdaq2.
Preparing to unpack .../3-libdaq2_2.0.4-3build2_amd64.deb ...
Unpacking libdaq2 (2.0.4-3build2) ...
Selecting previously unselected package libdumbnet1:amd64.
Preparing to unpack .../4-libdumbnet1_1.12-7build1_amd64.deb ...
Unpacking libdumbnet1:amd64 (1.12-7build1) ...
Selecting previously unselected package snort.
Preparing to unpack .../5-snort_2.9.7.0-5build1_amd64.deb ...
Unpacking snort (2.9.7.0-5build1) ...
Selecting previously unselected package oinkmaster.
Preparing to unpack .../6-oinkmaster_2.0-4_all.deb ...
Unpacking oinkmaster (2.0-4) ...
Setting up oinkmaster (2.0-4) ...
Setting up snort-common-libraries (2.9.7.0-5build1) ...
Setting up snort-common (2.9.7.0-5build1) ...
Setting up snort-rules-default (2.9.7.0-5build1) ...
Setting up libdaq2 (2.0.4-3build2) ...
Setting up libdumbnet1:amd64 (1.12-7build1) ...
Setting up snort (2.9.7.0-5build1) ...
Processing triggers for man-db (2.8.3-2) ...
Processing triggers for ureadahead (0.100.0-20) ...
ureadahead will be reprofiled on next reboot
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for systemd (237-3ubuntu10.57) ...
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ gedit /etc/snort/snort.conf
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo gedit /etc/snort/snort.conf

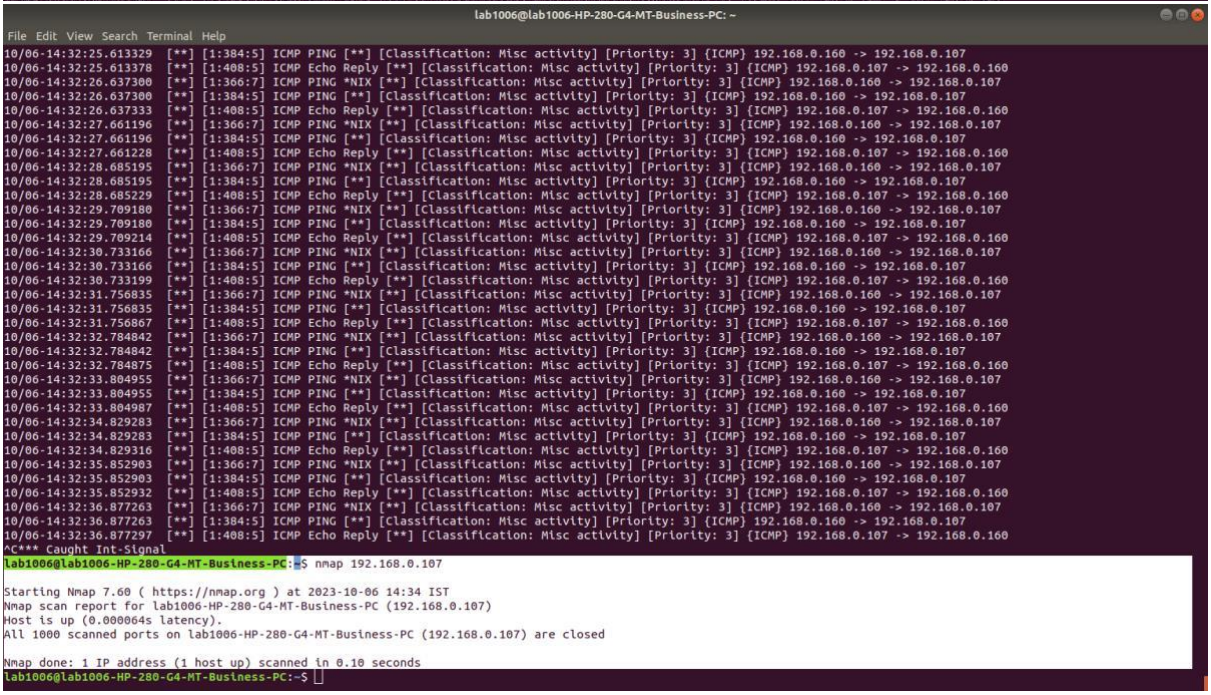

Activities
Terminal
Fri 14:05
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 227 bytes 23959 (23.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo apt-get install snort
[sudo] password for lab1006:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
libdaq2 libdumbnet1 oinkmaster snort-common snort-common-libraries snort-rules-default
Suggested packages:
snort-doc
The following NEW packages will be installed:
libdaq2 libdumbnet1 oinkmaster snort snort-common snort-common-libraries snort-rules-default
0 upgraded, 7 newly installed, 0 to remove and 340 not upgraded.
Need to get 0 B/1,424 kB of archives.
After this operation, 7,337 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Preconfiguring packages ...
Selecting previously unselected package snort-common-libraries.
(Reading database ... 162987 files and directories currently installed.)
Preparing to unpack .../0-snort-common-libraries_2.9.7.0-5build1_amd64.deb ...
Unpacking snort-common-libraries (2.9.7.0-5build1) ...
Selecting previously unselected package snort-rules-default.
Preparing to unpack .../1-snort-rules-default_2.9.7.0-5build1_all.deb ...
Unpacking snort-rules-default (2.9.7.0-5build1) ...
Selecting previously unselected package snort-common.
Preparing to unpack .../2-snort-common_2.9.7.0-5build1_all.deb ...
Unpacking snort-common (2.9.7.0-5build1) ...
Selecting previously unselected package libdaq2.
Preparing to unpack .../3-libdaq2_2.0.4-3build2_amd64.deb ...
Unpacking libdaq2 (2.0.4-3build2) ...
Selecting previously unselected package libdumbnet1:amd64.
Preparing to unpack .../4-libdumbnet1_1.12-7build1_amd64.deb ...
Unpacking libdumbnet1:amd64 (1.12-7build1) ...
Selecting previously unselected package snort.
Preparing to unpack .../5-snort_2.9.7.0-5build1_amd64.deb ...
Unpacking snort (2.9.7.0-5build1) ...
Selecting previously unselected package oinkmaster.
Preparing to unpack .../6-oinkmaster_2.0-4_all.deb ...
Unpacking oinkmaster (2.0-4) ...
Setting up oinkmaster (2.0-4) ...
Setting up snort-common-libraries (2.9.7.0-5build1) ...
Setting up snort-common (2.9.7.0-5build1) ...

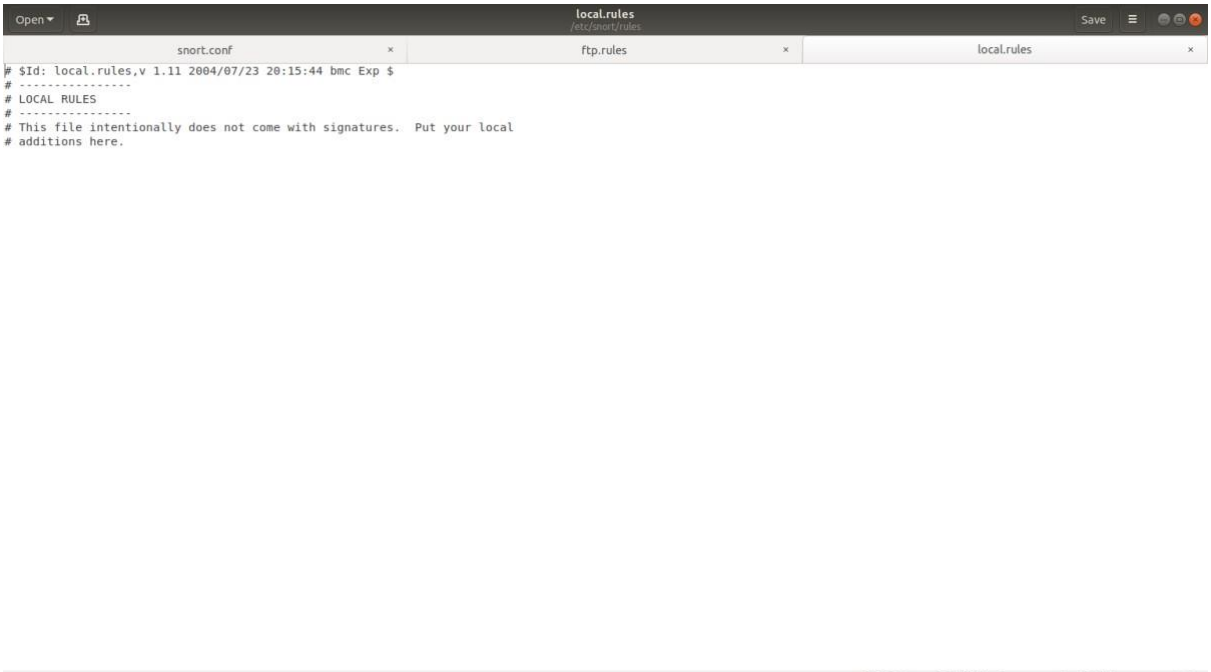

Activities
Terminal
Fri 14:02
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ ifconfig
enp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.0.107 netmask 255.255.255.0 broadcast 192.168.0.255
inet6 fe80::1593:a2b9:f028:7ee9 prefixlen 64 scopeid 0x20<link>
ether 04:0e:3c:19:2d:11 txqueuelen 1000 (Ethernet)
RX packets 5724 bytes 3064137 (3.0 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1478 bytes 133017 (133.0 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 227 bytes 23959 (23.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 227 bytes 23959 (23.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$

```
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo snort -T -c/etc/snort/snort.conf -i enp3s0
[sudo] password for lab1006:
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
ERROR: /etc/snort/snort.conf(51) Missing argument to HOME_NET
Fatal Error, Quitting..
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo snort -T -c/etc/snort/snort.conf -i enp3s0
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 800
0 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'SHELLCODE_PORTS' defined :  [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined :  [ 1024:65535 ]
PortVar 'SSH_PORTS' defined :  [ 22 ]
PortVar 'FTP_PORTS' defined :  [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined :  [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined :  [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2301 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510
7777 7779 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 555
55 ]
PortVar 'GTP_PORTS' defined :  [ 2123 2152 3386 ]
Detection:
   Search-Method = AC-Full-Q
    Split Any/Any group = enabled
    Search-Method-Optimizations = enabled
```

```
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
| State Density     : 10.6%
| Patterns          : 5055
| Match States      : 3855
| Memory (MB)       : 17.00
|   Patterns        : 0.51
|   Match Lists     : 1.02
|   DFA
|     1 byte states : 1.02
|     2 byte states : 14.05
|     4 byte states : 0.00
+----------------------------------------------------------------
[ Number of patterns truncated to 20 bytes: 1039 ]
pcap DAQ configured to passive.
Acquiring network traffic from "enp3s0".

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.8.1
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 2.4  <Build 1>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 9>
           Preprocessor Object: SF_GTP  Version 1.1  <Build 1>
           Preprocessor Object: SF_DNP3  Version 1.1  <Build 1>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 3>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 4>
           Preprocessor Object: SF_SIP  Version 1.1  <Build 1>
           Preprocessor Object: SF_IMAP  Version 1.0  <Build 1>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 13>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
           Preprocessor Object: SF_MODBUS  Version 1.1  <Build 1>
           Preprocessor Object: SF_REPUTATION  Version 1.1  <Build 1>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_POP  Version 1.0  <Build 1>
           Preprocessor Object: SF_SDF  Version 1.1  <Build 1>

Snort successfully validated the configuration!
Snort exiting
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
```

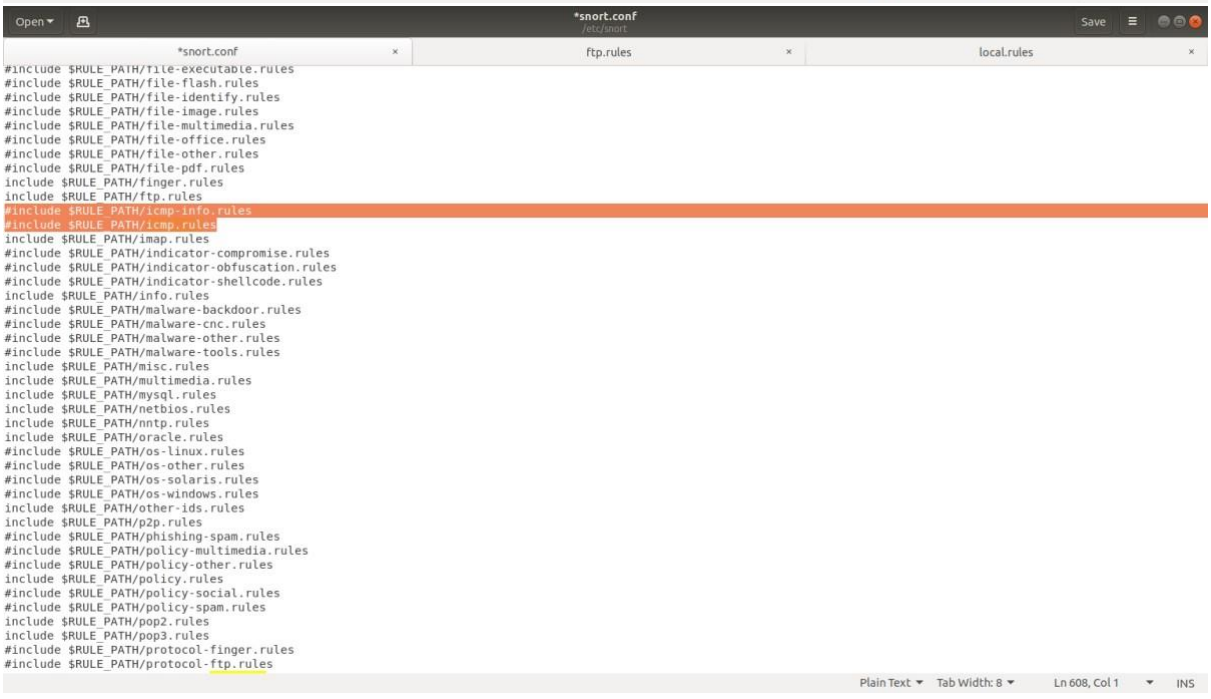

lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
Snort successfully validated the configuration!
Snort exiting
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i enp3s0
10/06-14:31:39.354328 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {IPV6-ICMP} :: -> ff02::16
10/06-14:31:39.370947 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
10/06-14:31:39.702377 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {IPV6-ICMP} :: -> ff02::16
10/06-14:31:39.766434 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {IPV6-ICMP} :: -> ff02::1:ff1a:5c74
10/06-14:31:42.117681 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
10/06-14:31:49.681063 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
10/06-14:32:01.256657 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
10/06-14:32:08.922515 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67
10/06-14:32:09.251847 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:09.251847 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:09.251877 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:10.253289 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:10.253289 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:10.253322 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:11.277408 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:11.277408 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:11.277438 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:12.301328 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:12.301328 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:12.301361 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:13.325410 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:13.325410 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:13.325442 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:14.349080 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:14.349080 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:14.349113 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:15.373367 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:15.373367 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:15.373399 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:16.397344 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:16.397344 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:16.397376 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:17.421337 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:17.421337 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:17.421370 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:18.445283 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:18.445283 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:18.445315 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:19.469269 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:19.469269 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:19.469303 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:19.888746 [**] [1:527:8] BAD-TRAFFIC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] {UDP} 0.0.0.0:68 -> 255.255.255.255:67

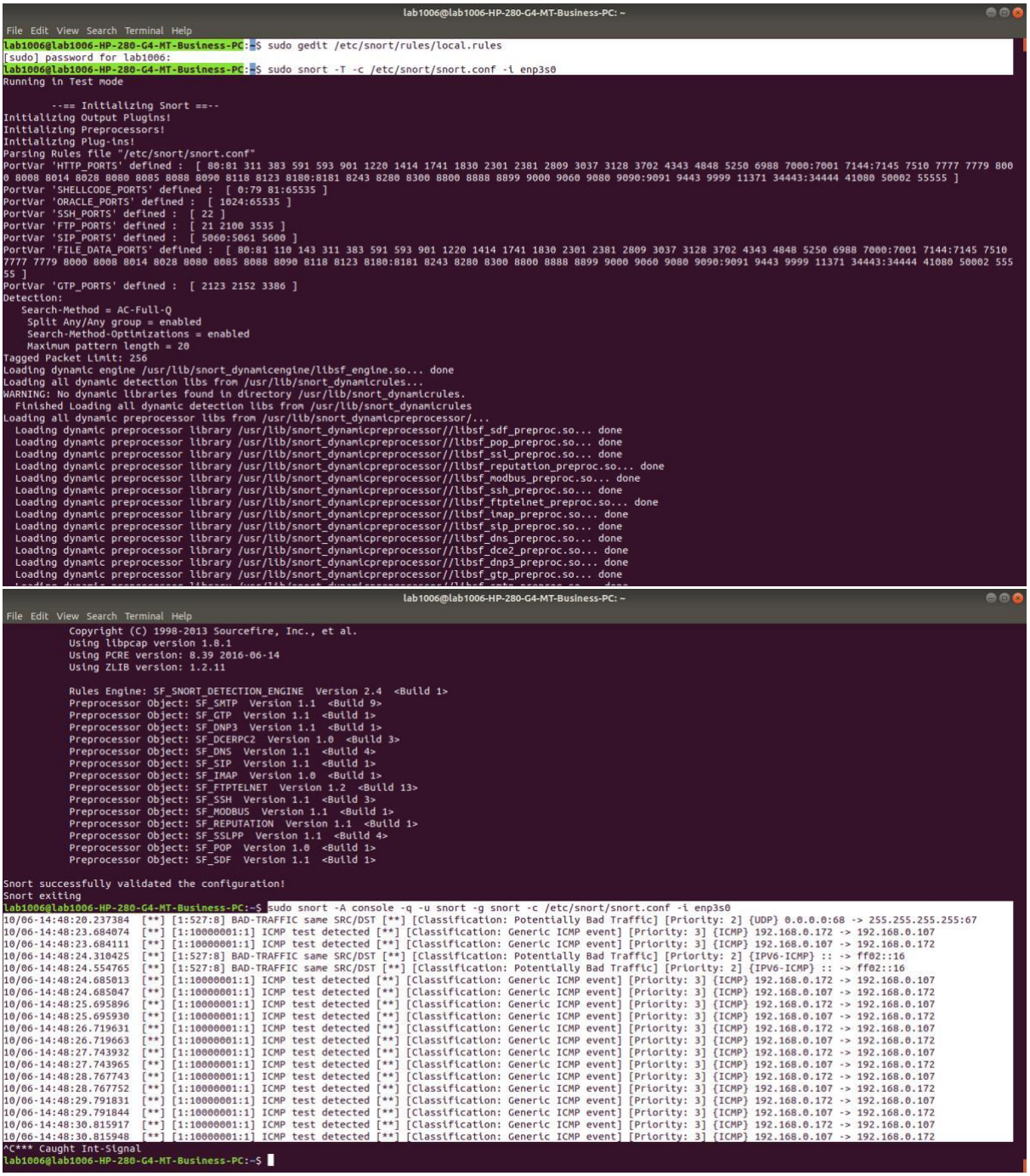

lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File Edit View Search Terminal Help
10/06-14:32:25.613329 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:25.613378 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:26.637300 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:26.637300 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:26.637333 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:27.661196 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:27.661196 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:27.661228 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:28.685195 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:28.685195 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:28.685229 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:29.709180 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:29.709180 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:29.709214 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:30.733166 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:30.733166 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:30.733199 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:31.756835 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:31.756835 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:31.756867 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:32.784842 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:32.784842 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:32.784875 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:33.804955 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:33.804955 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:33.804987 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:34.829283 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:34.829283 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:34.829316 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:35.852903 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:35.852903 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:35.852932 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
10/06-14:32:36.877263 [**] [1:366:7] ICMP PING *NIX [**] [Classification: Misc actlvity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:36.877263 [**] [1:384:5] ICMP PING [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.160 -> 192.168.0.107
10/06-14:32:36.877297 [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3] {ICMP} 192.168.0.107 -> 192.168.0.160
^C*** Caught Int-Signal
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ nmap 192.168.0.107
Starting Nmap 7.60 (https://nmap.org) at 2023-10-06 14:34 IST
Nmap scan report for lab1006-HP-280-G4-MT-Business-PC (192.168.0.107)
Host is up (0.000064s latency).
All 1000 scanned ports on lab1006-HP-280-G4-MT-Business-PC (192.168.0.107) are closed
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$


Open
local.rules
/etc/snort/rules
Save
snort.conf
ftp.rules
local.rules
$Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $

LOCAL RULES

This file intentionally does not come with signatures. Put your local
additions here.
Plain Text
Tab Width: 8
Ln 1, Col 1
INS


Open
*snort.conf
/etc/snort
Save
*snort.conf
ftp.rules
local.rules
#include $RULE_PATH/file-executable.rules
#include $RULE_PATH/file-flash.rules
#include $RULE_PATH/file-identify.rules
#include $RULE_PATH/file-image.rules
#include $RULE_PATH/file-multimedia.rules
#include $RULE_PATH/file-office.rules
#include $RULE_PATH/file-other.rules
#include $RULE_PATH/file-pdf.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
#include $RULE_PATH/icmp-info.rules
#include $RULE_PATH/icmp.rules
include $RULE_PATH/imap.rules
#include $RULE_PATH/indicator-compromise.rules
#include $RULE_PATH/indicator-obfuscation.rules
#include $RULE_PATH/indicator-shellcode.rules
include $RULE_PATH/info.rules
#include $RULE_PATH/malware-backdoor.rules
#include $RULE_PATH/malware-cnc.rules
#include $RULE_PATH/malware-other.rules
#include $RULE_PATH/malware-tools.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/multimedia.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/oracle.rules
#include $RULE_PATH/os-linux.rules
#include $RULE_PATH/os-other.rules
#include $RULE_PATH/os-solaris.rules
#include $RULE_PATH/os-windows.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/p2p.rules
#include $RULE_PATH/phishing-spam.rules
#include $RULE_PATH/policy-multimedia.rules
#include $RULE_PATH/policy-other.rules
include $RULE_PATH/policy.rules
#include $RULE_PATH/policy-social.rules
#include $RULE_PATH/policy-spam.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
#include $RULE_PATH/protocol-finger.rules
#include $RULE_PATH/protocol-ftp.rules
Plain Text
Tab Width: 8
Ln 608, Col 1
INS



Conclusion: In conclusion, this assignment involved the installation and configuration of Snort, a powerful Intrusion Detection System. By following the step-by-step instructions, we successfully installed Snort, edited its configuration file, and executed rules to detect ICMP activities. This hands-on experience enhanced our understanding of network security and IDS functionality.

Assignment 13

Aim: Explore the GPG Tool of linux to implement email security.

Lab Outcome: LO6

Theory:

1. What is Private Key Ring and Public Key Ring?

In the context of GPG (GNU Privacy Guard), a private key ring and a public key ring are essential components of the OpenPGP encryption and signing system. These key rings are used for managing cryptographic keys for secure communication.

- **Private Key Ring:** This is a collection of private keys owned by a user. Private keys are used for decrypting messages sent to you and for signing messages to ensure their authenticity. Each user typically has their private key ring, which should be kept confidential and protected at all costs. Only the owner of the private key ring should have access to it.

- **Public Key Ring:** This is a collection of public keys, which are meant to be shared openly. Public keys are used by others to encrypt messages meant for you and to verify the digital signatures you create with your private key. Public keys are freely distributed and can be obtained from a keyserver or directly from the person they belong to.

2. Write the commands used for key generation, export and import of keys and signing and encrypting the message in gpg tool.

Key Generation:
To generate private and public key pairs for sender and receiver, you can use the following commands:
gpg --gen-key

Export a Public Key:
To export your public key, which can be shared with others, use the --export command and redirect the output to a file:

gpg --output my_public_key.gpg --export your_email@example.com

Import a Public Key:
To import someone else's public key, use the --import command:

gpg --import their_public_key.gpg

Create a file containing sender's private key:

gpg –export -secret-key -a username>filename

Import the public key of the receiver:

gpg --import filename_containing_public_key_of_receiver

Signing Keys :
Sender can sign the public key of the receiver to establish trust:
gpg --sign-key receiver_email

Encrypting Data :
Encrypt a file for a specific receiver:
gpg --encrypt -r receiver_email name_of_file .gpg file created

Encrypt and sign a file (ASCII format):

gpg --encrypt --sign --armor -r receiver_email name_of_file ASCII file created

Encrypt and sign a file (.gpg format):

gpg --encrypt --sign -r receiver_email name_of_file .gpg file created

Decrypting Data :
Decrypt a file:

gpg -o myfiledecrypted -d myfile.txt.gpg

Output:

```
teaflame@LAPTOP-P2FVIJDD: $ gpg --gen-key
gpg (GnuPG) 2.2.19; Copyright (C) 2019 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/teaflame/.gnupg' created
gpg: keybox '/home/teaflame/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: athar
Email address: athar@gmail.com
You selected this USER-ID:
    "athar <athar@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/teaflame/.gnupg/trustdb.gpg: trustdb created
gpg: key A023142A64B2E0C6 marked as ultimately trusted
gpg: directory '/home/teaflame/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/teaflame/.gnupg/openpgp-revocs.d/F552545DFF20C09D99C15685A023142A64B2E0C6.rev'
public and secret key created and signed.

pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      F552545DFF20C09D99C15685A023142A64B2E0C6
uid                    athar <athar@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]
```

```
teaflame@LAPTOP-P2FVIJDD:~$ gpg --gen-key
gpg (GnuPG) 2.2.19; Copyright (C) 2019 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name: gaobl
Email address: gaobl@gmail.com
You selected this USER-ID:
    "gaobl <gaobl@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key D9D3DF8A0D24318C marked as ultimately trusted
gpg: revocation certificate stored as '/home/teaflame/.gnupg/openpgp-revocs.d/8BE4092A007F24ECC5B96708D9D3DF8A0D24318C.rev'
public and secret key created and signed.

pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      8BE4092A007F24ECC5B96708D9D3DF8A0D24318C
uid                      gaobl <gaobl@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]
```

```
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export -a athar>spub
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export-secret-key -a athar>spri
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export-secret-key -a athar spri
-----BEGIN PGP PRIVATE KEY BLOCK-----

lQWGBGUsF0gBDADBbOmQGg5c76y8CcZrSwfjaoegoQX4Y0+QZL99Cyv33ES3KOUW
afPRUysRx7RvzLP2f+lGB6dYUML6l+l8617jCrok26/4n5PkZlOZ1hZWUMqFLpZA
+2ccmUqg+QiP6bJGVzruihI/pPFjCoP/+T/+CjYwnAvsXIHb8xdor5XRA/bCmosw
D+sXKaxvSnrhbty6NDwo11eecARqR7E0b6TyjgNRwYxHXE5NTTgUEjXCAEjxPRqo
xNNJg07hKoD0NY9tKk8MFNz42Ij4jLrwHn12d3L63d9nL2C2zof2VuXWezeYzvLB
N8zVJ/QNmOPmOXSNN7fOnlAjeTAnpNHTVPdVs/LZxd4NOftzhK70TlSa5Za8+YLV
WEisf8/MNd5Tjv292Awc7lawD3NPptAvGtVVff10JWUFlJrBy6t+yPOvke0w6Kgy
H4JDnBCMsXYWHiOZeOx/MkkkftXcnpSVRm7xxBPrA5LyfsjzaouXbFtHaJUWX3n3
r1qlQXBMQRsfx/cAEQEAAf4HAwJu7QoqNozcwf9pBcgOapnSASWDUyVU6VS2FAJY
GuY6QVhVp7LKPaOlItJk2pnaRh6dbrfJilg4lAsfP6RAXJSQXzc+YijWPTqUuuAX
qdXTIWVX1mCMNDbdFuOQ7/Z97THurr8a5cvbtahpc8btKutxUm78qHN2ZA1aAGlm
BiRUd4N/VC2+V7m4uhrEYeo4egPiNJw2dvQ+Nf9lraWR//vWqxb/Znk5HFiYK2Tb
AJk8jlG+p6xwc43ZMN7oIw4KohDEpMWu4545Yg665iAWtUXg1ipKqJoW9UlXsmrl
LfuBPPHUYwU4atMJuVXAKvBLPp64Ai38RFIYLvmWw4hzB81ce/s69TGouUk+aKaf
kJh+jaFFBZKOUAupCU8q9X2LVHeXIWvuHOjX2lvqbSkpgb11vl6tYV3WdAfU4qJh
jYTb/cGzri59GcDPajS7Jhl1hgHRZszCHVZJlMvIQtZz6C7Gm+K3DSnR28UwzA9y
KiBnxWO3xKvAjFscOvHDhWU6LjSumpBic69I9AF/RXDDBvOpJoWJO7F6/k3SSPCE
znCngH2T1nwPm07ANw9q4t7ffwBYwfTIE4EuoJwXqk8sS+CQ4sUzvVlb9NSwpsjV
FVOMSKeVESOeA7z+6XuNGH8LjiHJtz4ylAMlN6A3Ektbp7oy4ud978pbzbTK7a5k
Dh2+Zl6NqWgNS2GQj2j4SvMvLkj0IC56jfcqbVxfOUsUmLtNU3red6/A7+h93Mux
vluafnkFxDUYV2k4bm3raxrTTAwHGMqYWA23F284noj1h9Zo4ycPFsiezybnS0Nm
IcBjvTlFsJWIMJltoaBqc2XiGM0cKAd2jvyQs+EigvPqRDfjZ9TF3aVHRRKOcdXF
eQgu+dVz+6eE7ekhrb6b9y6HzJXW5wYi44CLPgWizuWPrI4w348TuFyrG8thqTEx
jWxayC4qO4Z7PoYd/JNs8hn7LCgtA9yai/c14xnbrwMDCchOJq8MsS44qiRwU4Jp
88VJ56vWZ9t1Z0SE4QPttueTkF5kPJuQ6K2y7uOhu6am6UWz6P68UdZ0fxO+Kgop
XSbGTkIk3qYrrp0+kdNXTrxIZ/Hn+sFJklDLdwy8lk8TiM0q6Y/nSOgo7whXiNNC
LfC1FFHTl3V+pA5bKJFOptFDe+5JrrmYEabNEbRLt9wNoX6T/h4szcJxXUaabLGS
gYbmYr9O3kLrmttV8W8kxQAz2omMa+XeniNlI5ToQYRDrjAIDLQYagAgE0R49hng
S3SAF8t0AZaLVMxznvROWypTNfPt9XITVBZ3IoblSE5QfpfulSTlgl5M3kwHPpwb
jj0JfBtOgXb+4m0lQ3DocACYCr0/QcOwbrQXYXRoYXIgPGF0aGFyQGdtYWlsLmNv
```

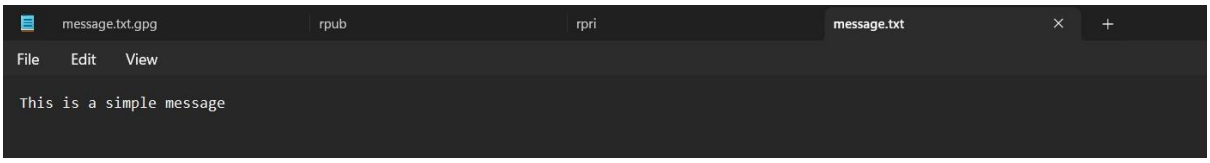
```
pOsndy0LOeGLmgIhVPqNcScQQ4aO8J5CUazIGQSPE0GP+HuxUtZ2dO4KmfR7HMa8
Ae4KJQARAQAB/gcDAhaVAimVpUmn/ytT5sizWKnBG2FCnRwMVJAWGbqZQu8YS5ng
XPuqtvO4BAwEMnklbszmZd2SnqneIASIdppNrcIfP+y8vnbhlVMgLcSwUsH20EDC
ewSKO3+F6nr5b8wU/dme2LRCtCdhE/pGv/I2ZWWpj98X88V5bIzQs7rXEr+7ej4s
Mzc9rfhvE9BoEfOq/5RJ9rLeL9rWZCDiOuwnxj2mmpiRT0moDANH7Jj1pbULAONx
cLiCwFL5tP8vu7y+8Npvzx4abZ8fEoKBDBInlF8A7SY9kGLGuepgtiJQWJ97w7YE
MBoBmFeiZJryronJopZzYd5pYxOqUu111UN9sPx8qQf8F1S7xensvN7IYCQ8y09O
wQQwUkWjHgqMcUBh4zrysIAyxWhiez8oPudcPusycistyR7AUBkCNUmiXijg2cMn
wr19Ai/fjPlR5tX+n3Q9lmbnmhecYIHwXhD181SU9WV0vowyaxnjhTexAuhkJcT5
xka2RFd4xIz2CbIk6kloPDlwamUj2UJMvgFhr84tO+o3HQwDUR+8mdwKae3fIpmo
NsVZhRhs9Zb60J1Knnfl45MqwnZeRhXqeF0CSv3aePKgg82Lo/CyT9a2MNtnKfkJ
1lvX2HgObMHugBDwIeHCBR0ikMXgkkTjY1NIGEFB6HiMWdCkItEygqLaJUuyhQK/
f60nYn7bXjsMJC3hELS4Zjs/BeenPmDz2d41zGDpFEEvfTB3IgK0oH56KvPvOo0Z
Cp5HV05Dsa0/p7tyhQTfoYg4of0+QaJF3d6XpbSRDpnUV2MS2Q5ng4KnINGdPJTL
MrP/1OaPpD3jfzj8sGpa2zGgAJKLQknqKTlMPN5L+NThSO2jugUBNGPKKuWVSIDv
54ra66Wt0ajNxHgyCP9rUG/NDlOSxKGifwEr9or0Y5gFQquu/MsGGE4MdWMW4vUO
w1DNUhCZB94Ucs08rK9Yy0Nq1oWj9a64enjTAaOIHWl9CToq7WNG8eSqafmDScUu
kvZK5jQcybF8fq0z0+wJjMoqFAgi5eQAJU7YEwHlziw9ixAMJ2bfbOhqJrrxpqF0
a+y0lqW1ab5vrucJDQPzNpDOcCg2oYzOef0jATCPkyTAZlqw3FSV+kDRWguwKAo1
o1nArBH8cLwobYtrDpvPyMmKtfaKjgVToAINEb9NZi5UpsavrQ592LuQSUMkqC81
tpW4KVlvgdP9C78dnxVQitA7aQqi3QWYdatD0KzXUPV8mN4UvWtqZaGmbJ4f1gCP
V2xkN+uLcgtWiwhnFYp1AD4SnF6WdfvgSaDt7qahfUX+0BgqWCDicOkinXPsIRUy
dWvE2R0HZs+XC0hSvLc5lYT2iQG8BBgBCgAmFiEE9VJUXf8gwJ2ZwVaFoCMUKmSy
4MYFAmUsF0gCGwwFCQPCZwAACgkQoCMUKmSy4MYOgAwAwSX4pW93pZO9fsU0n4aR
DaT2k1fVOdrtYUbWugFxyiK/3DqKdqNUpfAOPl/1hWt3dEUipH3Cip+NA0FMO9Y7
aCNL8Y39cuZsCLqgfrzfMvJAtz/ni5KerD+HbPJb9/FORBw+L+43oPC9c2WeKRCE
X8PEcdxSlZNnJMLQmZf3bbUwWGgNClihTzr5Lb9r/EChFjWsrFu4YUaWM4mCueyu
w5SkoHfErIfDwCiv6j9peqgErf0LArKg80X5EZZp3qft4KST0rp6Ce0S3zI/lSZj
Xfb0aRXp5ffgn3ylPtqoN4SYXf4PCvMZE9iTqbXB0ugIN2fumhoHe2lsvlH2RJ0P
Jr1ttY4CLmuQBzPMsk1aBL++cdhRtbzS8spLEJRVn34KZIpCrW4KFOVYbQ+pvB5Z
liq/+DTSJ0d1SeALmoYb8zHE2Vf2yHaibNx6Q4E0pa+/H1iPolTj6OBzS1MlO/AF
PGpVnwCFV59m1yHRSQEyBAfXZnUX12RdqxDRhigAoWnf
=bRxr
-----END PGP PRIVATE KEY BLOCK-----
teaflame@LAPTOP-P2FVIJDD:~$
```

```
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export -a gaobl>rpub
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export-secret-key -a gaobl>rpri
teaflame@LAPTOP-P2FVIJDD:~$ gpg --export-secret-key -a gaobl rpri
-----BEGIN PGP PRIVATE KEY BLOCK-----

lQWGBGUsGSIBDAC6bYldDbnCZqgzgJuqJtiHSNG88IxLMAoP4pWU0QtxV6R7cVxm
jr4U6qIQHZZgaJ+5V6bmpWVPiy3moioJlz5/nowp0ISVreyCo4xW+h1YY0e8nD1P
ipSiLnk1fuNC5L0rOrcLiMvrv0LgduzYTx26TexrXHJKfsreAiuIPfuTNvooSV3J
VKZcWFFuPbME8w06bh67kEM69AKVFnXpixpKZCkiIzkkOw1aJt51BneNXJoKJL+6
vKoi8sjfXBJxX7AEGeGp26HwztC+gqhBEtX5ePhe2SY+5tmzRqb5I2DMnmpJpkzP
HAc56Hs4B7kX793kpufU8miMKFmB7WzkR0NnkjAo8Djt7TBByfdyPNvXG+Tz2nCW
QcUeaqQnki0IpOIgeCDnvO/PsgojB6YTtih/6cIGgA1l+VQlCDdMD2M41rzjUrTy
/fBomGy28wkR4wMQ/c2kPB47NEUQQA2zP9nJMU/tU1+4ewuO/csnwWpyBnDaLL30
NKV291dMW9sVVi8AEQEAAf4HAwIPK70Zm88FLf9NtfWy3K7PifQ01gIx0ssFDS2z
FODMuRugTesVPh1+yrhoB9BzQALuHMkbayXEfaZVmvwp6tuAbd159dBBrJOSPh7d
aEHt7Y6tnWsr7m8sk+MS4kiJvufyV8ankEgdzUmdZfZgTMx+x1iIKybkUtEvcMf2
qiORxq2uxXfTPCNkWHeptg2oc57vpA2cRO8YyHaCbTmF+xpKQ2/Tp6njvZyDiV82
lV/0b8svbQcHtZLKeQIsC+3BsqoXe2lSTyGIKzZK3eVpo05bZisMBXZfRzPalNQj
qWx5Oq9KGUCm4xfUZAPOt6Qy+Tetoh7CU4ei7B5Ppjx89sTOZJLDTyRTYTdWEMxV
iKFAG4uhjAfeVVUJjFHdqy9k5aBv6NRNmtuZPXHZohPfbpTr/0+RVQvVqrVI3JXW
HhGIKQNvoF+1IZRFGALZGeAhZV9ySx2K3RizHuJGIlleyIUYMmY6NegFEf9ydbpK
4LUxr8feRc6xeCh/VFk3n7z5kPhRub68kT4lK2r8XFUD+UK96xSX55lmUP0mmBct
Lw2TjeB4w3iot6MtrX7GXlj1m0uJ/SCfJ38HMW6Z2uurGHryDophiOfKSBn3OB0u
TZw0/+B7s7TwKVGanpX/A7C1ZTxKLVmYn5pTgwnDAA+o02nmTFIdeV3OGwtZm1FQ
6BxVOjqjqtbPbI/ICRRboLbySIuS/aUqM85JNNhtaBhGu0OYZBPe3lORrcYuJv2C
r8xeDEUbrOEI0B0F6QKI8BlAEkoDcAxuULVwEOekE9j2gqKYdik/JspdLIyf/a88
s2xSl3Py6LbrET0paqLRR1gk6tmTJtcIGktMGPszK8pKhewQIU7lz+3QCTzFINu/
C5DWLPGkn1Fb5c4hsGZLrAJqtV97lqH5m1+x09OZCw+TjWz2hBQyM8WofRQNSCX6
dp9e0wezqnn5sjGp7IpcjrXGuYEeZawV/ATDJWqM9/JzAMkqIrLnYNrFDMdKCsiM
lkr+Sa24DRcuDTwSySg1f70A38yejhLaDkCXXY7g6RFdS36TDuuDJ43FWZNYRmlW
Uc7ffZTRsXN0h1XA3ZcFtSWhpq51tagwoS8TMMvusKG25AOFw+1xa2BMGsR+8OND
tXvuQGkRLOCqS6siIKBsj7IgEk2Yf0K37O+iTY7BISKXkOZzT4JdzzIIyyaRTK7q
fXIusHFc9iyMuTIMvKOVDhpHC3T4Jgl+7mZhqgG/7Br9aegCKzJ2KudjH/MP0Ijy
d3cVbk1op4PJ1UGsX1sMMg6gYisq3C7bexdvyqWOz9cyv3F35xlUvv7Vvh2ZpFJJ
eWWpdkxB75NzftMoGijl9h0mSA0VK2lspLQXZ2FvYmwgPGdhb2JsQGdtYWlsLmNv
```

```
LoawiEgWNR345/NnmTOwbZFK3VKEFf/jAOLrQ4hyGxbvcfLh+9clUTEVnJdO/XXy
3nIviwARAQAB/gcDAom2iS6wObcw/xBNqxGZCGMbOwms9YhuN4W6ThT5qZvPsQeh
JFlTXZtH//M4Q6z8MvhE1u1ozq3/Sp4F54MpQWBeJS/W1rwV/MyLCUexYwI/0Pnp
cSzuKxEerOT233Fdb0Izhxi9ca/4OXiss19Xei6lfwgVGeXa16z7n1hEguHOnRvf
Ejlhwj/rROs5s4CorUMJD8dfE8UTo4NsLnyeuKthg5R9pKvd1EXPBzC3ODbjNJVs
TmBqNt3uFRmLEeoi+AiFJ0JRF/up9yCtmVQvPdXkwwxj4NwBbfQBJrcMK7DqqwET
hluoDuJHkzXWizCEojfEp2w8lwynIUnB5MZGBLL81yP/jfH0X0MQrvlkV/l4bEbG
VJu9Yn1hk50wmdTIgv1EUMdvUrL1VNplI1s2PI+sDDwhJ+W2HYbgNoF0jRFt/KHr
gdYHDJru9LpNr73DmYWWUQGQ+CqZw+vm2AvyFsRU+PXvynsX+uiNDCsjM6hsmnDY
XqSYZ2EsMJEcLFKFuEVYLyughS3a5PDU2TPPNM9NUx7uuM6b8hzbCQMxX6fVmXzR
6KxDb0wbpIpOKC9AKkSMz40XS9IOKlxyu0k7k6JfPKmnOS4LuVBLQsgALQVcXf7A
FiPi2He3GfDihgxlrfLrwFRbip9PL7gHBHjWpGfT9uhoVm+UzQUChuIqumtJsCUI
a7SyYlWv1FDDmL8zFy7zd3BZQiwlPMnekwlnTgo7X/5d/wJGW+xy4sJDBxSCYmN+
riy2ro6aj0IUsQFYxe4COn2GxgVGIpYpRzZb0uqK9LeIba0LtWgQS+fA5otQ/zQF
tWuUDh3QBerSFcHNn5CAeP882ssacNXh0ohl0foEgooUT8vME3QqL9KGVaIO3ReB
gl/2zws7MHh86OFpHOHDxxYSFQr/FOSIRPprbRsGX71F1KS7ERPkl7sFIQzrcCoI
7Yl8i6xcI6C1K2xalwP4EmjnvlBi9zOj8SF4NhsLR+W+dP2/PIHXkl/AmeHwz61M
8w3y9K679j7iB9K9px8XesaCK3vton5iUV3B9Gkd4YtLs7JkB3XIR5ZPM5Ooe5oV
F3IC14QMhfNN33E7BZM+taZks9dIW+HPig13QRref4i3pXJlTLJgUx624D5c1GMY
hJ+TfiRxyuLQA5hGs4q7IZ+kWxVd2Nac4UJPY+9YfevT4NHIFqeysf/8+Cvdl+XS
OmAywzrHLahNO2CBQCWJ6SwVAGnsNbMMwrg8Qxvbnz1RH+rCERJqkg1l4PtHlmOf
0Ge7s7C87BonDOV1Bns1wmvFoyW/KXbLrAR3SiSoAC6xBf6HebWCUFRV72ULhjsM
O6aPEwZ2IIYFuG5nNvns1kGWiQG8BBgBCgAmFiEEi+QJKgB/JOzFuWcI2dPfig0k
MYwFAmUsGSICGwwFCQPCZwAACgkQ2dPfig0kMYws3wwAnxaZAiRqZ0Msv9NAS+kx
+A6jPDyQu5DjZob1piTR5M69Wh36kUnDUpCbLkF+Vwu76L2YzA0sDg5ZfpFQAP19
pwPhI0FXP2hXjKoN4WjsfUoTK+4QEteDXlqDL4lLKNnsh+VyP2pXL3syXgBSUrO8
960Z2ePe8mcGfBZ7iADitWjMv+7qnss5IeCPdyqv2hcITQymQ+O0FN6Up0r0RNj+
RkfwlsldzQa+NA4ufcKt5MTEe6SUep+2S7h6HfYdsbWqpFBdrP8lMTXyl0U1T+KE
qi/QIJ+8COV95sxGL7rOjzgepwOqqHpIiuZWImL9LNYgu0lLh/NfRanSFpa0hFTp
XiB+Q2qHsngXIiy6cLtqpdxRzp0WWz7+s1m7Apk/AXJWAYScvBAd4c9A/BSKw7ly
c6bgls32ZiWe7IgZdwcROjZhpLcG3SnazhP4WMTdtDhXXYKAmWyS+gCO60r3iM6O
IOLDKDICBpI3rBPf8PYWNSOCFvAyePjQ4c2IWSUceIM2
=0Xot
-----END PGP PRIVATE KEY BLOCK-----
teaflame@LAPTOP-P2FVIJDD:~$
```

```
PGpVnwCFV59m1yHRSQEyBAfXZnUX12RdqxDRhigAoWnf
=bRxr
-----END PGP PRIVATE KEY BLOCK-----
teaflame@LAPTOP-P2FVIJDD:~$ gpg --import rpub
gpg: key D9D3DF8A0D24318C: "gaobl <gaobl@gmail.com>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
teaflame@LAPTOP-P2FVIJDD:~$ gpg --encrypt -r gaobl hi
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: next trustdb check due at 2025-10-14
gpg: can't open 'hi': No such file or directory
gpg: hi: encryption failed: No such file or directory
teaflame@LAPTOP-P2FVIJDD:~$ gpg --allow-secret-key-import --import rpri
gpg: key D9D3DF8A0D24318C: "gaobl <gaobl@gmail.com>" not changed
gpg: key D9D3DF8A0D24318C: secret key imported
gpg: Total number processed: 1
gpg:              unchanged: 1
gpg:       secret keys read: 1
gpg:  secret keys unchanged: 1
teaflame@LAPTOP-P2FVIJDD:~$ gpg --list-keys
/home/teaflame/.gnupg/pubring.kbx
---------------------------------
pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      F552545DFF20C09D99C15685A023142A64B2E0C6
uid           [ultimate] athar <athar@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]

pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      8BE4092A007F24ECC5B96708D9D3DF8A0D24318C
uid           [ultimate] gaobl <gaobl@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]
```

```
teaflame@LAPTOP-P2FVIJDD:~$ gpg --list-keys
/home/teaflame/.gnupg/pubring.kbx
---------------------------------
pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      F552545DFF20C09D99C15685A023142A64B2E0C6
uid           [ultimate] athar <athar@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]

pub   rsa3072 2023-10-15 [SC] [expires: 2025-10-14]
      8BE4092A007F24ECC5B96708D9D3DF8A0D24318C
uid           [ultimate] gaobl <gaobl@gmail.com>
sub   rsa3072 2023-10-15 [E] [expires: 2025-10-14]

teaflame@LAPTOP-P2FVIJDD:~$ nano msg.txt
teaflame@LAPTOP-P2FVIJDD:~$ nano message.txt
teaflame@LAPTOP-P2FVIJDD:~$ gpg -e -u athar -r gaobl message.txt
teaflame@LAPTOP-P2FVIJDD:~$
```



Conclusion:

In summary, we explored GPG's private and public key rings, key management, and security processes. These are vital for secure communication and trust verification in digital exchanges.

Animesh Parab T2-T21 88

Written Assignment 1

1. Explain the padding scheme used in RSA. Why it is used? What is its limitation? (LO2)

Ans :-

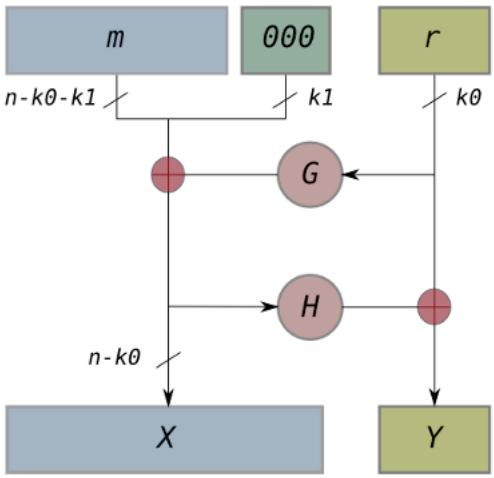
Padding in RSA (Rivest-Shamir-Adleman) is a crucial aspect of the encryption and decryption process. RSA padding schemes are used to add additional bits to the plaintext before encryption and remove them after decryption. Padding is primarily used to address certain security vulnerabilities and limitations of the RSA algorithm.

In cryptography, padding is a number of operations including appending data to anywhere of the plaintext before encryption. The purpose of a padding scheme is to avoid adversary to retrieve information of the primitive, for example, a chosen plaintext attack or an adaptive chosen ciphertext attack in RSA. Optimal Asymmetric Encryption Padding Optimal Asymmetric Encryption Padding(OAEP) was invented by Mihir Bellare and Phillip Rogaway in 1994 and enchanted by Don Johnson and Stephen Matyas in 1996. It was standardized as RSAES-OAEP in PKCS#1 Version 2 and lately republished as RFC 2437. OAEP combined with RSA is good at performance and provides good security especially against adaptive chosen ciphertext attack.

There are two aims of OAEP:

A. Adding random padding to plaintext can convert RSA from a deterministic scheme into a probabilistic one.

B. Prevent leaking any encryption structure information caused by chosen plaintext attack. The padding process of OAEP is shown as below:



Where

• n: the length of bits of RSA modulus

• k0 and k1: numbers defined by OAEP protocol

• m: the plaintext with a length of n−k0−k1 bits

• G and H are two cryptographic hash functions

• ⊕: xor operation

• r: a random generated string of k0 bits

Encoding of OAEP:

a) The plaintext m is padded with k1 zeros appending m to m’ with n−k0 bits length.

b) r is converted into a string of n−k0 bits by a cryptographic hash function G.

c) X=m’⊕G(r).

d) X is reduced to k0 bits by H.

e) Y=r⊕H(X).

f) The result of padding is X and Y.

Decoding of OAEP:

a) r is recovered by r=Y⊕H(X).

b) m’ is recovered by m’=X⊕G(r).

Security of OAEP

The OAEP provides semantic security against chosen ciphertext attack, though Victor Shoup raised doubt about whether OAEP could provide such security. In 2001, Eiichiro Fujisaki’s team proved that RSA-OAEP is semantical secure in the random oracle model.

Both block ciphers and RSA are permutations on a block(RSA's block isn't an integral number of bytes), so it's clear that both of them need some kind of padding if the data size doesn't correspond to the block size.

With block ciphers the padding doesn't do much: It fills up the remainder of the block, and tells you how much padding there was.

With RSA the padding is essential for its core function. RSA has a lot of mathematical structure, which leads to weaknesses. Using correct padding prevents those weaknesses.

For example RSA Encryption padding is randomized, ensuring that the same message encrypted multiple times looks different each time. It also avoids other weaknesses, such as encrypting the same message using different RSA keys leaking the message, or an attacker creating messages derived from some other ciphertexts.

RSA padding should always be used, and it has a minimum size of dozens of bytes, as opposed to a single byte with most block cipher paddings.

Why Padding is used in RSA

Padding schemes in RSA (Rivest-Shamir-Adleman) encryption are used to address certain vulnerabilities and limitations associated with the basic RSA algorithm. The most commonly used padding schemes in RSA are PKCS#1 v1.5 padding and OAEP (Optimal Asymmetric Encryption Padding). These padding schemes serve several important purposes:

1. Security: RSA encryption without padding can be vulnerable to attacks like the padding oracle attack, which can reveal information about the plaintext. Padding schemes add randomness and structure to the plaintext before encryption, making it harder for attackers to exploit vulnerabilities.

2. Data Integrity: Padding schemes ensure that the encrypted message can be decrypted correctly. They help distinguish between valid and invalid ciphertexts, preventing errors or tampering during transmission.

3. Preventing Attacks: Padding schemes prevent certain mathematical attacks on the RSA algorithm. Without padding, an attacker could potentially recover the plaintext by analyzing the ciphertext and exploiting patterns in the encryption process.

4. **Randomness**: Padding schemes often include random bytes, adding a level of randomness
to the encryption process. This randomness helps to ensure that encrypting the same plaintext multiple times results in different ciphertexts, improving security.

Two commonly used padding schemes in RSA:

1. PKCS#1 v1.5 Padding:

- PKCS#1 v1.5 padding is an older padding scheme used with RSA encryption.
- It involves adding a specific sequence of bytes to the plaintext before encryption.
- This padding includes a block type byte, random padding bytes, and a message digest.

- PKCS#1 v1.5 padding is still widely supported but is considered less secure than OAEP.

2. OAEP (Optimal Asymmetric Encryption Padding):

- OAEP is a more modern and secure padding scheme.
- It uses a hash function and a random number generator to add padding to the plaintext.
- OAEP padding is designed to provide better security against various cryptographic attacks, including chosen ciphertext attacks.
- It ensures that each encrypted message is unique, reducing the risk of patterns that could be exploited by attackers.

Limitations of Padding in RSA:-

While padding schemes in RSA enhance security, they also come with limitations:

1. Padding Overhead: Padding increases the plaintext's length, resulting in a larger ciphertext. This overhead can be a concern when transmitting data efficiently or when storage space is limited.

2. Compatibility: Different padding schemes exist, and the choice of padding can impact interoperability between different implementations of RSA. It's essential to use a padding scheme that is compatible with the recipient's decryption algorithm.

3. Padding Oracle Attacks: Some padding schemes, like PKCS#1 v1.5, are vulnerable to padding oracle attacks if not implemented correctly. These attacks can leak information about the plaintext.

4. Vulnerabilities : Some padding schemes have been vulnerable to specific attacks. For example, the PKCS#1 v1.5 padding scheme had vulnerabilities in the past that led to practical attacks like the Bleichenbacher attack.

5. Complexity: Implementing padding schemes correctly can be complex and error-prone. Errors in the padding process can lead to security vulnerabilities.

Let's illustrate PKCS#1 v1.5 padding with a simple example using Python:

//python

```
from cryptography.hazmat.primitives import serialization
from cryptography.hazmat.primitives.asymmetric import padding
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import rsa
# Generate RSA keys (usually done once)
private_key = rsa.generate_private_key(
public_exponent=65537,
key_size=2048
)
public_key = private_key.public_key()
# Message to be encrypted
message = b"Hello, RSA Padding!"
# Encrypt with PKCS#1 v1.5 padding
ciphertext = public_key.encrypt(
message,
padding.PKCS1v15()
)
# Decrypt with PKCS#1 v1.5 padding
decrypted_message = private_key.decrypt(
ciphertext,
padding.PKCS1v15()
)
print(f"Ciphertext: {ciphertext.hex()}")
print(f"Decrypted Message: {decrypted_message.decode()}")
```

In this example, we generate an RSA key pair, encrypt a message using PKCS#1 v1.5 padding,
and then decrypt it. PKCS#1 v1.5 padding ensures that the plaintext length is the same as the RSA modulus size before encryption.

Animesh Parab 88 T2---T21

Assignment 2

An Intrusion Detection System (IDS) is a security technology used to monitor network or system activities for suspicious or malicious behavior and to identify potential security threats. IDSs are crucial for maintaining the integrity and security of computer networks and systems. There are several types of IDSs, each with its own approach to detecting intrusions. Below, I'll explain the different types of IDSs and their working, along with their advantages and limitations:

Signature-Based IDS (Network-Based and Host-Based):

- **Definition:** Signature-based IDSs rely on predefined patterns or signatures to identify known threats or attacks.
- **Working:**
- **Network-Based Signature IDS:** It inspects network traffic, such as packets and network flows, searching for patterns that match known attack signatures. For example, if the IDS identifies a packet with a signature matching a known virus, it raises an alert.
- **Host-Based Signature IDS:** This type monitors activities on individual hosts or endpoints, such as servers or workstations. It reviews system logs, file changes, and processes to detect known malicious patterns.
- **Advantages:**
- Effective at detecting well-known threats.
- Low false positive rates as it's looking for specific patterns.
- Efficient in terms of resource usage.
- **Limitations:**
- Ineffective against zero-day attacks because they lack known signatures.
- Regular signature updates are necessary to remain effective.
- Limited to identifying threats based on established patterns.
- **Example:** A network-based signature IDS might identify and alert the IT team to a known pattern of SQL injection attempts in incoming web traffic.

2. **Anomaly-Based IDS (Network-Based and Host-Based):**

- **Definition:** Anomaly-based IDSs establish a baseline of normal behavior and then flag any deviations from this baseline as potential intrusions.
- **Working:**
- **Network-Based Anomaly IDS:** It learns the typical patterns of network traffic and raises alerts when it detects significant deviations from the norm, such as unusual data volume or unusual port usage.
- **Host-Based Anomaly IDS:** This type observes the behavior of individual systems or users. It tracks file access, login times, and system processes, and alarms if any behavior significantly deviates from what's typical.

- **Advantages:**
- Can detect zero-day attacks and previously unseen threats.
- Adapts to changing attack patterns.
- Effective against insider threats as it can detect unusual internal behavior.
- **Limitations:**
- Higher false positive rates, as any deviation from the baseline may trigger alerts.
- Requires substantial historical data to establish accurate baselines.
- Might not catch sophisticated attacks that mimic normal behavior well.
- **Example:** A host-based anomaly IDS could raise an alert when a user, who typically logs in at 9 AM and logs out at 6 PM, suddenly starts accessing the system at midnight.

3. **Hybrid IDS:**
   - **Definition:** Hybrid IDS combines elements of signature-based and anomalybased detection to enhance overall threat detection capabilities.
   - **Working:** Hybrid IDS simultaneously employs both signature-based and anomaly-based techniques. It looks for known attack patterns and deviations from normal behavior.
   - **Advantages:**
   - Balances the strengths of signature-based and anomaly-based IDS.
   - Offers a more comprehensive security posture.
   - **Limitations:**
   - It can be resource-intensive due to running both types of analysis.
   - Still vulnerable to false positives and false negatives, albeit potentially to a lesser extent.
   - **Example:** A hybrid IDS might detect a known malware signature in an incoming packet and also raise an alert if that packet contains unusual or unexpected data patterns.

4. **Behavior-Based IDS:**
   - **Definition:** Behavior-based IDS focuses on the behavior of software applications and processes to identify deviations from expected behavior.
   - **Working:** It builds profiles of how applications and processes should behave. When it detects significant changes or actions inconsistent with the established profiles, it triggers an alert.
   - **Advantages:**
   - Effective against advanced persistent threats (APTs) and other complex attacks.
   - Less reliant on specific attack signatures.
   - **Limitations:**
   - Requires significant computational resources to profile and analyze behavior.
   - May generate false positives when legitimate software behavior changes.

- **Example:** A behavior-based IDS might alert administrators when a web server, which typically only handles incoming requests, suddenly initiates unauthorized outbound network connections, indicating a potential breach.

Each type of IDS has its place in a comprehensive security strategy. Organizations often use a combination of these IDS types to maximize threat detection while minimizing false alarms and vulnerabilities.

Machine learning

In Cyber Security

Team Member

Anurag Pareek (89)
Animesh Parab (88)

1. Introduction

5 min

➢ What Is Machine Learning in Security?

- **Machine learning (ML)** lets computers learn without being explicitly programmed. Put another way, machine learning teaches computers to do what people do: learn by experience. Machine learning is a domain within the broader field of artificial intelligence.

- In **security**, machine learning continuously learns by analyzing data to find patterns so we can better detect malware in encrypted traffic, find insider threats, predict where “bad neighborhoods” are online to keep people safe when browsing, or protect data in the cloud by uncovering suspicious user behavior.

2. How ML Works

15 min

➢ How does machine learning work in security?

- The cyber threat landscape forces organizations to constantly track and correlate millions of external and internal data points across their infrastructure and users. It simply is not feasible to manage this volume of information with only a team of people.
- This is where machine learning shines, because it can recognize patterns and predict threats in massive data sets, all at machine speed. By automating the analysis, cyber teams can rapidly detect threats and isolate situations that need deeper human analysis.

➢ How does it work?

- The details of machine learning can seem intimidating to non-data scientists, so let's look at some key terms.
- Supervised learning calls on sets of training data, called “ground truth,” which are correct question-and-answer pairs. This training helps classifiers, the workhorses of machine learning analysis, to accurately categorize observations. It also helps algorithms, used to organize and orient classifiers, successfully analyze new data in the real world.
- An everyday example is recognizing faces in online photos: Classifiers analyze the data patterns they are trained on--not the actual noses or eyes--in order to correctly tag a unique face amongst many millions of online photos.

3. How machine learning helps security...

30 min

Find threats on a network	Machine learning detects threats by constantly monitoring the behavior of the network for anomalies. Machine learning engines process massive amounts of data in near real time to discover critical incidents. These techniques allow for the detection of insider threats, unknown malware, and policy violations.
Keep people safe when browsing	Machine learning can predict “bad neighborhoods” online to help prevent people from connecting to malicious websites. Machine learning analyzes Internet activity to automatically identify attack infrastructures staged for current and emergent threats.
Provide endpoint malware protection	Algorithms can detect never-before-seen malware that is trying to run on endpoints. It identifies new malicious files and activity based on the attributes and behaviors of known malware.
Protect data in the cloud	Machine learning can protect productivity by analyzing suspicious cloud app login activity, detecting location-based anomalies, and conducting IP reputation analysis to identify threats and risks in cloud apps and platforms.
Detect malware in encrypted traffic	Machine learning can detect malware in encrypted traffic by analyzing encrypted traffic data elements in common network telemetry. Rather than decrypting, machine learning algorithms pinpoint malicious patterns to find threats hidden with encryption.

3 Types of Machine Learning in Cybersecurity

There are three types of machine learning used in cybersecurity: supervised learning, unsupervised learning and reinforcement learning.

SUPERVISED LEARNING

Supervised learning involves training an algorithm on labeled data, so it learns how to organize data based on the relationships between inputs and outputs. Human guidance is often needed to assist algorithms during training. Machine learning algorithms use supervised learning to classify data as neutral or harmful, identifying threats like denial-of-service attacks and predicting future cyber attacks.

UNSUPERVISED LEARNING

Unsupervised learning refers to an algorithm trained on unlabeled or raw data, and it labels and classifies data without human guidance. Security teams rely on unsupervised learning to train algorithms to detect new and more complicated cyber attacks, especially as hackers develop different techniques to infiltrate company defenses.

REINFORCEMENT LEARNING

Reinforcement learning is a trial-and-error approach where an algorithm learns new tasks by being punished for incorrect actions and rewarded for correct ones. In cybersecurity, machine learning algorithms use this technique to improve their ability to detect a wider range of cyber attacks. Teams can also employ reinforcement learning to automate repetitive tasks, resulting in more efficient IT and security processes.

How Is Machine Learning Used in Cybersecurity?

A subset of artificial intelligence, machine learning uses algorithms born of previous datasets and statistical analysis to make assumptions about a computer's behavior. The computer can then adjust its actions, even performing functions it wasn't programmed to do. These abilities have made machine learning a crucial cybersecurity asset.

HOW IS MACHINE LEARNING USED IN CYBERSECURITY?

- Detecting threats in early stages
- Uncovering network vulnerabilities
- Reducing IT workloads and costs

DETECTING THREATS IN EARLY STAGES

With its ability to sort through millions of files and identify potentially hazardous ones, machine learning is increasingly used to uncover threats and squash them before they can wreak havoc.

Software from Microsoft showcased this skill in 2018, when cybercrooks attempted to infect over 400,000 users with a cryptocurrency miner during a 12-hour time frame. The attack was stopped by Microsoft's Windows Defender, a software that employs multiple layers of machine learning to identify and block perceived threats. The crypto miners were shut down almost as soon as they started digging.

UNCOVERING NETWORK VULNERABILITIES

Rather than wait for cyber attacks to happen, companies are taking a more proactive approach with machine learning. Penetration testing involves simulating a cyber attack to locate weak points in a company's networks, firewalls and systems. Machine learning can execute this task and apply software patches, code fixes and other solutions to address any holes in an organization's security suite.

In addition, machine learning's ability to learn from historical data allows it to pick up on unusual software and user behavior during these kinds of training sessions. The technology then remembers how specific cyber attacks occur and can determine which ones pose the biggest threats based on a network's vulnerabilities.

Benefits of Machine Learning in Cybersecurity

With its range of applications, machine learning offers many advantages to IT and security personnel.

AUTOMATED CYBERSECURITY PROCESSES

Machine learning can learn new functions and get better at performing existing ones on its own, resulting in automated workflows. Security and IT teams can then leave basic responsibilities to machine learning while focusing their time and resources on addressing new cyber threats, fixing urgent flaws and completing other advanced tasks.

STRENGTHENED SECURITY PROCEDURES

Reviewing a company's security infrastructure, machine learning algorithms can expose weak points, recommend fixes and help teams prepare for a variety of cyber attacks. This way, security and IT teams can address threats before they even happen, establishing the procedures and systems needed to fend off more complex attacks.

ABILITY TO HANDLE LARGE DATA SETS

Humans may struggle to deal with large volumes of data, but machine learning can quickly process and analyze larger data sets. Algorithms can spot trends much faster than humans and alert teams of developing cyber attacks. IT and security personnel can then take immediate action, snuffing out cyber attacks in their early stages before they spread.

Machine Learning in Cybersecurity Challenges

While machine learning in cybersecurity meets various IT and security needs for businesses, the technology must continue to adapt to an ever-changing digital ecosystem. Even then, machine learning may not be able to overcome some limitations and outside factors.

INCREASING NUMBER OF CONNECTIONS

The number of connected devices is expected to reach 29 billion by 2027 as hybrid and cloud environments become more popular. Company networks are constantly adding new computers, tablets and other devices, putting pressure on machine learning to account for and protect more connections against cyber attacks.

SOCIAL ENGINEERING SCHEMES

Not even the strongest machine learning-based security system can make up for human error. Social engineering strategies like phishing emails take advantage of relationships built on trust and authority. If teams aren't trained to identify these schemes, companies may fall victim to a socially engineered cyber attack.

TECH TALENT SHORTAGES

Despite IT and security being essential for companies in the digital age, more than 85 million skilled jobs are expected to go unfilled by 2030. Companies need data scientists and IT workers who know how to maintain machine learning algorithms and interpret their analyses. Without this kind of literacy, teams may struggle to adopt ML-based cybersecurity solutions.

MACHINE LEARNING DATA NEEDS

Machine learning depends on large amounts of historical data to detect patterns that it can apply to future situations. The problem is that machine learning cybersecurity data isn't common. And any existing security data may be considered sensitive material, so teams might have to get creative when finding data to train machine learning algorithms.