Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./~~Miss~~ Niranjan Rajesh Joshi

of IT Department, Semester V with

Roll No. 51 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

20/10/23

Teacher In-Charge

Head of the Department

Date 20/10/2023

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
1)	Breaking shift cipher and mono alphabetic substitution cipher using frequency analysis method	1 - 10	19/07/23	
2)	Cryptanalysis or decoding of polyalphabetic cipher	11-20	26/07/23	
3)	Block cipher modes of operation using Advanced Encryption Standard	21-32	09/08/23	
4)	Implementation and Analysis of RSA cryptosystem and Digital signature scheme using RSA	33-42	26/07/23	
5)	To explore Hashdeep tool in Kali Linux generating, matching and auditing	43-62	02/08/23	
6)	Study use of network reconaissance tools like WHOIS, dig, traceroute, nslookup, dmitry, nikto to gather information about networks	63-74	02/08/23	
7)	Study packet sniffer tools wireshark and TCPDump	75-83	09/08/23	
8)	Installation of NMAP and using it with different options to scan open ports. perform OS finger printing, ping scan, TCP port scan	84-98	23/08/23	
9)	Simulate DOS attack using HPING3	99-105	06/09/23	
10)	To study and configure firewall using IPtables		27/09/23	
11)	Installing snort, setting in IDM and writing rules for Intrusion detection	106-113	06/09/23	
12)	Explore GPG tool of Linux to implement email security	114-125	13/09/23	
	Written Assignment 1	126-128	20/09/23	
	Written Assignment 2	129-138	30/09/23	
	Final security lab presentation	~~129-138~~	18/10/23	

Shah 20/10/23

Key exchange
Kerberos
DOS - types
DDOS
AES , DES - full diagram
IP tables - detail
Knapsack protocol
Modes of operation AES
Hashing - MD5 , SHA with diagram

Lab Outcomes:

LO 1: Illustrate symmetric cryptography by implementing classical ciphers.
LO 2: Demonstrate Key management, distribution and user authentication.
LO 3: Explore the different network reconnaissance tools to gather information about networks.
LO 4: Use tools like sniffers, port scanners and other related tools for analyzing packets in a network.
LO 5: Use open-source tools to scan the network for vulnerabilities and simulate attacks.
LO 6: Demonstrate the network security system using open source tools.

Roll Number:- 2205051
Name:-Niranjan Joshi
Date:- 19/07/2023

Lab Assignment No:-1

Aim:- Breaking Shift Cipher and Monoalphabetic substitution cipher using frequency analysis method.

Lab Outcome Attained :- LO1

Theory:-

What is Shift Cipher ? (with eg)

Shift Cipher is one of the earliest and the simplest cryptosystems. A given plaintext is encrypted into a ciphertext by shifting each letter of the given plaintext by n positions.An example of encrypting the plaintext by shifing each letter by 3 places.

Plaintext: shift cipher is simple
Ciphertext: vkliwflskhulvvlpsoh

x be the position number of a letter from the alphabet

n be the an integer which 0<=n<=25, it is the key for encryption and decryption of shift cipher cryptosystem

The encryption process is (the x here represents a letter from plaintext):
x+n (mod 26)

The decryption process is (the x here represents a letter from ciphertext):
x-n (mod 26)

How and why shift cipher can broken using Brute force attack ?

A brute force attack is a method of breaking an encryption by trying every possible key until the plaintext is revealed. In the case of a shift cipher, there are only 26 possible keys, so a brute force attack is relatively easy to implement.

To break a shift cipher using a brute force attack, the attacker would simply encrypt the ciphertext with each of the 26 possible keys. Once the plaintext is revealed, the attacker would know which key was the correct one.

For example, if the ciphertext is "FEQJP" and the shift is 3, then the plaintext would be "hello". The attacker would simply encrypt the ciphertext with each of the 26 possible keys until they get "hello" as the plaintext.

What is monoalphabetic cipher ?(with eg)

A monoalphabetic cipher is a type of substitution cipher in which each letter of the plaintext is mapped to a single letter of the ciphertext. This means that there is a one-to-one correspondence between the letters of the plaintext and the letters of the ciphertext.

The Caesar cipher is a simple monoalphabetic cipher that shifts each letter of the plaintext by a fixed number of positions. For example, if the shift is 3, then the letter A would be encrypted as D, B would be encrypted as E, and so on.

Here is an example of how the Caesar cipher would work:

Plaintext: HELLO Key: 3 Ciphertext: KILO

To decrypt the ciphertext, simply shift each letter back by the same number of positions. In this case, you would shift each letter back 3 positions.

Ciphertext: KILO Key: 3 Plaintext: HELLO

Can monoalphabetic cipher broken using brute force attack ? Why ?

No , monoalphabetic cipher cannot be broken using brute force attack because of its large keyspace . The keyspace of a monoalphabetic cipher is the set of all possible keys that can be used to encrypt a message. In the case of a monoalphabetic cipher, the key is simply a permutation of the alphabet. This means that there are 26! possible keys for a monoalphabetic cipher with a 26-letter alphabet.

Frequency analysis can be used to break monoalphabetic cipher

How can it be broken using frequency analysis attack ?

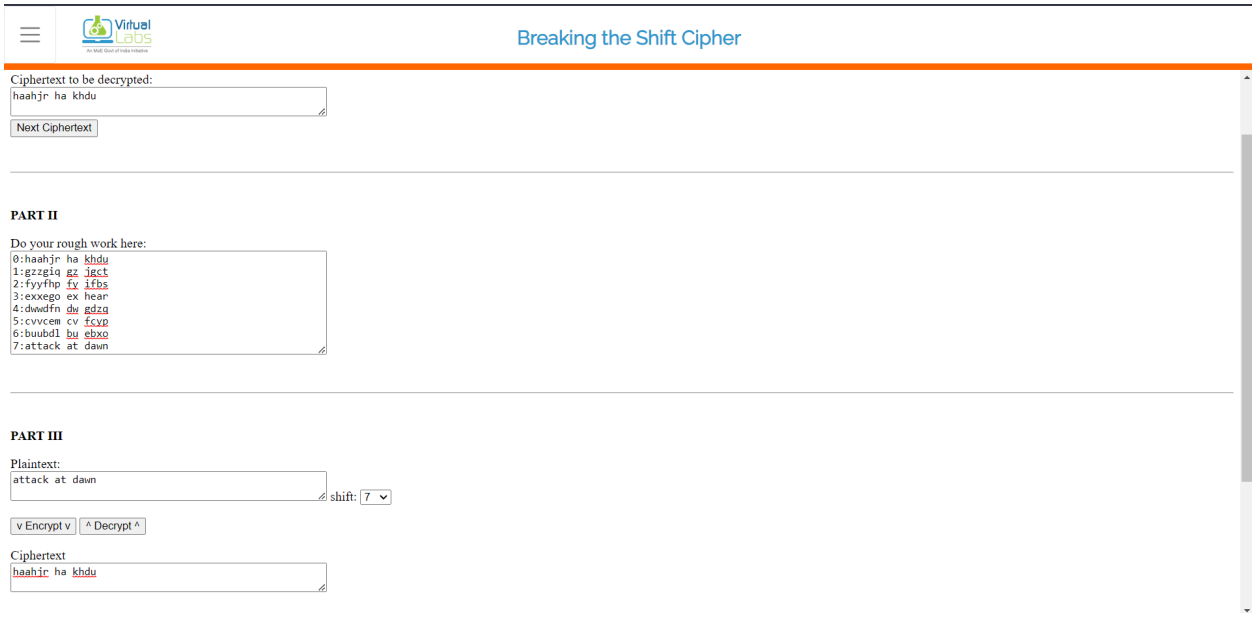
Frequency analysis is a cryptanalysis technique that exploits the fact that certain letters are more common than others in a language. For example, the letter "E" is the most common letter in the English language, followed by "T" and "A". This means that if you know the frequency of letters in the plaintext, you can use frequency analysis to guess the key of the ciphertext.

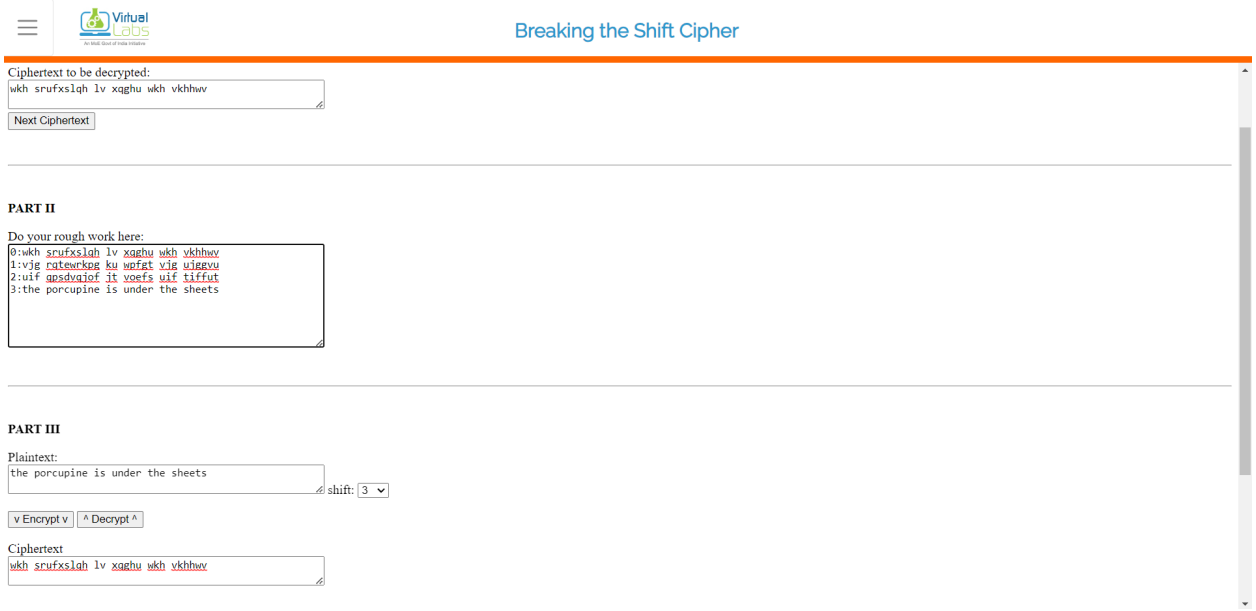
To break a monoalphabetic cipher using frequency analysis, you would first need to create a frequency table of the letters in the ciphertext. This table would show how often each letter appears in the ciphertext. Once you have created the frequency table, you would then need to compare it to the frequency table of the plaintext.

If the two frequency tables are similar, then you can use the frequency table of the plaintext to guess the key of the ciphertext. For example, if the letter "E" appears most often in the ciphertext, then you can guess that the letter "E" in the plaintext is mapped to the letter "E" in the ciphertext.This procedure can be repeated and then ciphertext can be converted to plain text .

Output Screenshot:-

1)breaking the shift cipher





PART III

Plaintext:

```
attack at dawn
```

shift: 7

v Encrypt v | ^ Decrypt ^

Ciphertext

```
haahjr ha khdu
```

PART IV

Enter your solution Plaintext and shift key here:

```
attack at dawn
```

Key 7

Check my answer!

```
CORRECT!!
```

PART III

Plaintext:

the porcupine is under the sheets

shift: 3

v Encrypt v ^ Decrypt ^

Ciphertext

wkh srufxslqh lv xqghu wkh vkhhwv

PART IV

Enter your solution Plaintext and shift key here:

the porcupine is under the sheets

Key 3

Check my answer!

CORRECT!!

2)breaking the monoalphabetic substitution cipher - Q1

Breaking the Mono-alphabetic Substitution Cipher

PART I

Decrypt the following cipher text. A tool to simulate the Mono-Alphabetic Subsitution cipher is provided beneath for your assistance.

Here is the table of frequencies of English alphabets for your reference:

a	b	c	d	e	f	g	h	i	j	k	l	m
8.167	1.49	2.782	4.253	12.702	2.228	2.015	6.094	6.966	0.153	0.772	4.025	2.406
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
6.749	7.507	1.929	0.095	5.987	6.327	9.056	2.758	0.978	2.360	0.150	1.974	0.074

```
dkxyvrh 1 - qegt vkr hxccwv keur: xuwdr wn cehrq nwvvwtp et vkr
hwsrhcxto gwvk krh nwnvrh, gkrt nkr tevwdrn x vxuowtp, duevkrq gkwvr
hxccwv gwvk x yedorv gxvdk hit yxnv. nkr leuuegn wv qegt x hxccwv keur
gkrt niqqrtub nkr lxuun x uetp gxb ve x dihwein kxuu gwvk fxtb uedorq
qeehn el xuu nwmrn. nkr lwtqn x nfxuu orb ve x qeeh vee nfxuu leh krh
ve lwv, civ vkheipk gkwdk nkr nrrn xt xvvhxdvwsr pxhqrt. nkr vkrt
```

Next Ciphertext

Calculate Frequencies in ciphertext

Ciphertext Frequencies:

a	b	c	d	e	f	g	h	i	j	k	l	m
0.000	1.037	2.282	3.942	8.091	1.452	3.112	5.602	2.075	0.000	8.506	1.452	0.415
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
7.469	1.867	1.452	3.32	11.618	0.622	4.979	5.602	9.959	6.639	7.884	0.622	0.000

Breaking the Mono-alphabetic Substitution Cipher

PART II

Note that the *ciphertext 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 *insensitive* function and replaces only cipher text (lower case) by plain text (upper case):

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

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!!

3)breaking the monoalphabetic substitution cipher - Q2

Breaking the Mono-alphabetic Substitution Cipher

PART I

Decrypt the following cipher text. A tool to simulate the Mono-Alphabetic Subsitution cipher is provided beneath for your assistance.

Here is the table of frequencies of English alphabets for your reference:

a	b	c	d	e	f	g	h	i	j	k	l	m
8.167	1.49	2.782	4.253	12.702	2.228	2.015	6.094	6.966	0.153	0.772	4.025	2.406
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
6.749	7.507	1.929	0.095	5.987	6.327	9.056	2.758	0.978	2.360	0.150	1.974	0.074

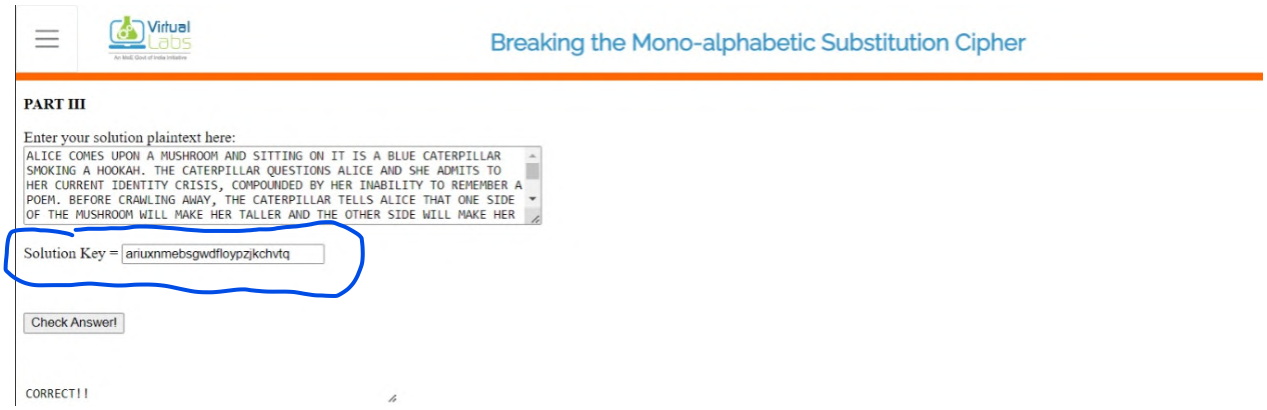
```
zelpjxp. zex rpxagz lnn jhl obxixz npld jex dkzeplld. lfx zbux dagxz
exp zepbfg zdawwxp jeaf xcxp, hebwx afljexp iakzxz exp fxig jl mplh
ebme bfjl jex jpxxz, hexpx a obmxlf dbzjagxz exp nlp a zxpoxfj. hbje
zldx xnnlpj, awbix rpbfmz expzxwn raig jl exp kzkaw exbmej. zex
zjkdrwxz kolf a zdaww xzjajx afu kzxz jex dkzeplld jl pxaie a dlpx
aooplopbajx exbmej.
```

Next Ciphertext

Calculate Frequencies in ciphertext

Ciphertext Frequencies:

a	b	c	d	e	f	g	h	i	j	k	l	m
8.084	6.854	0.176	4.394	7.206	4.394	1.757	1.582	2.988	7.909	2.636	7.381	1.582
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
1.582	2.285	8.26	0.000	1.582	0.000	0.703	1.933	0.000	5.097	14.06	0.176	7.381



Conclusion:-

Implemented shift cipher and monoalphabetic substitution cipher on random plain text and hence learned more about symmetric cryptography and its functioning

Roll Number:- 2205051
Name:-Niranjan Joshi
Date:- 26/07/2023

Lab Assignment No:-2

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

Lab Outcome Attained :- LO1

Theory:-

How vigenere cipher works?(with eg)

The Vigenere Cipher is a method of encrypting and decrypting messages. It is a polyalphabetic substitution cipher, which means that it uses multiple cipher alphabets to encrypt the plaintext.

The Vigenere Cipher operates by using a keyword, typically a word or phrase, as the basis for encryption. The keyword is repeated to match the length of the plaintext message. Each letter in the keyword is then used to determine the shift value for the corresponding letter in the plaintext.

To encrypt a message, each letter of the plaintext is shifted by the corresponding letter in the keyword. Here the plaintext is divided into group of n characters each where n is length of key used for encryption . For Eg:-

Plaintext :- hello world

Key:- vig

Plain text will be divided as follows and mapped with key given accordingly :- (in group of 3)

Hel low orl d

Vig vig vig v

Then **(p+c)%26** formula is applied and letter corresponding to it is encoded character

Where , **p:- number corresponding to character in plain text starting from zero , c:-number corresponding to character present in key starting from zero**

So applying above text we get **cipher text** as:- **cmrgwcjzry**

Explain in brief how Kasiki test is used to break vigenere cipher?

The Kasiski test is a cryptanalytic method for breaking the Vigenere cipher. It works by looking for repeated sequences of characters in the ciphertext. If a repeated sequence of characters is found, the distance between the occurrences of the sequence is likely to be a multiple of the length of the keyword.

For example, suppose the ciphertext contains the sequence "ABCABC". If the keyword is 3 characters long, then the distance between the occurrences of the sequence will be 3, 6, 9, 12, etc. If the keyword is 4 characters long, then the distance between the occurrences of the sequence will be 4, 8, 12, 16, etc.

By finding the distances between repeated sequences in the ciphertext, the cryptanalyst can narrow down the possible values of the keyword length. Once the keyword length is known, the cryptanalyst can then use other methods to break the cipher.

Here is an example of how the Kasiski test can be used to break a Vigenere cipher:

Ciphertext:

ABCDEFGH

HIJKLMNOPQRSTUVWXYZ

The cryptanalyst first finds a repeated sequence in the ciphertext. In this case, the sequence "ABC" is repeated twice. The distance between the occurrences of the sequence is 12.

Since the distance between the occurrences of the sequence is a multiple of 3, the cryptanalyst knows that the keyword length is 3. The cryptanalyst can then use other methods to break the cipher, such as frequency analysis.

The Kasiski test is a simple but effective method for breaking the Vigenère cipher. It is not foolproof, but it can be used to break ciphers that have been encrypted with short keywords.

How Playfair Cipher works ? (with eg)

The Playfair cipher is a symmetric encryption technique that employs a 5x5 square matrix (keyed matrix) of letters to encrypt and decrypt messages. It was invented by Charles Wheatstone in 1854 and later popularized by Lord Playfair. The main idea behind the Playfair cipher is to encrypt pairs of letters (digraphs) from the plaintext into ciphertext using the following rules for encryption :

1)Before encrypting plain text if two consecutive letters in plaintext are same then insert bogus character ‘x’ in between them

2)If both characters in pair are in same row replace them by immediate right character from same row

3)f two characters in pair appear in same column replace them with immediate bottom character

4)If above two cases are not satisfied replace them by character in same row but in column of other character

Eg:-

Suppose we want to encrypt the message "HELLO" using the keyword "KEYWORD" (without repeating letters, and 'J' is combined with 'I').

Keyed Matrix Setup:

K E Y W O

R D A B C

F G H I L

M N P Q S

T U V X Z

The plaintext "HELLO" is divided into digraphs: "HE" and "LLO".

Encryption Rules:

"HE": H and E are in the same row, so we replace H with the letter to its right (E) and E with the letter to its right (F).

"LLO": L and O form a rectangle, so we replace L with the letter at the opposite corner of the rectangle (M) and O with the letter at the opposite corner of the rectangle (N).

Ciphertext:

The encrypted message is "FE MN NM."

To decrypt the ciphertext, the recipient would use the same keyed matrix and apply the decryption rules in reverse.

How cryptanalysis on playfair cipher can be done ?

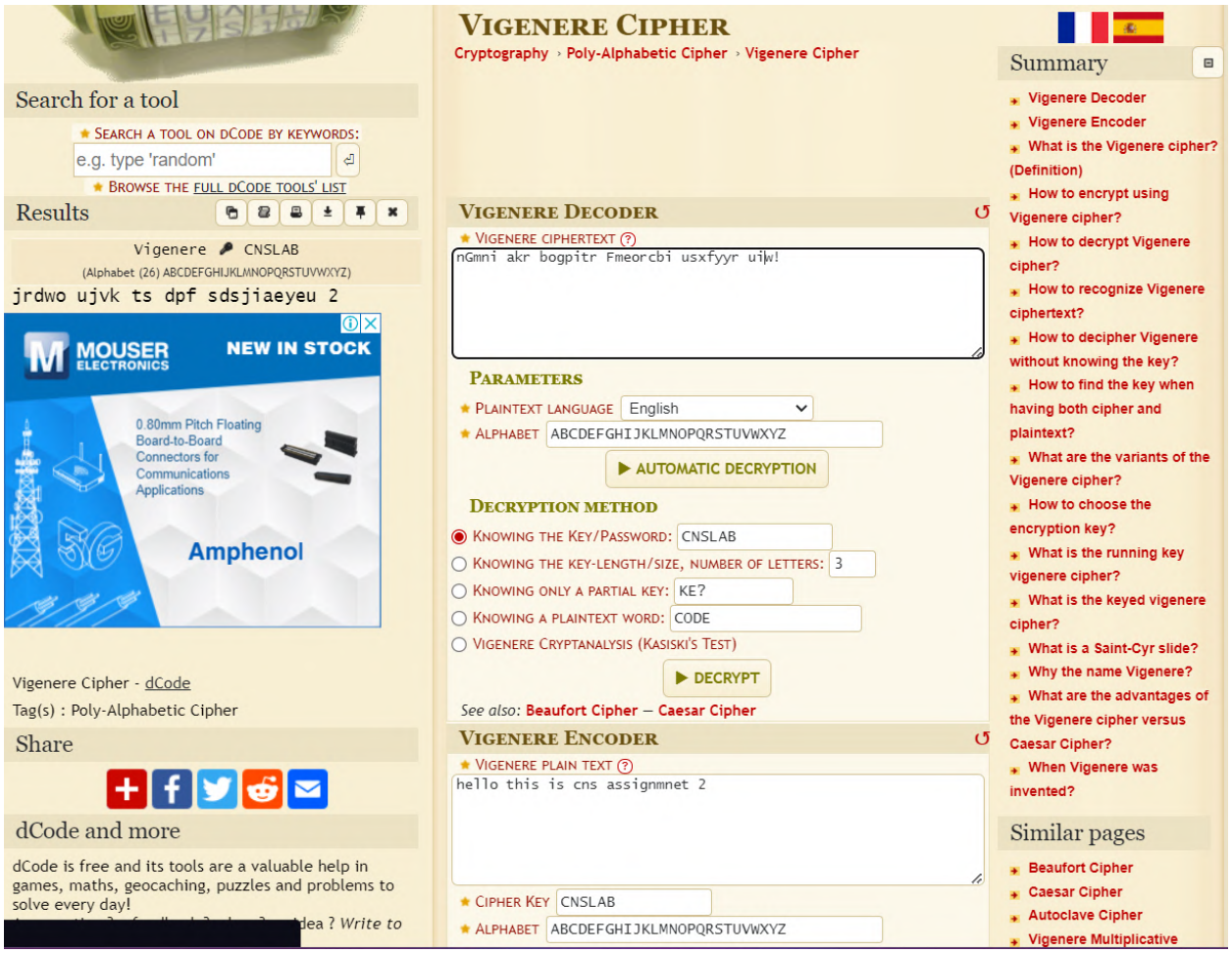
Cryptanalysis on the Playfair cipher involves attempting to break the encryption without knowing the key or the plaintext-ciphertext pair. Most common way is using **frequency analysis**

Cryptanalysts can perform frequency analysis on the ciphertext to identify patterns in the letter distribution. In English text, certain letters appear more frequently than others (e.g., 'E' is the most common letter). By analyzing the frequency of letters in the ciphertext, they can make

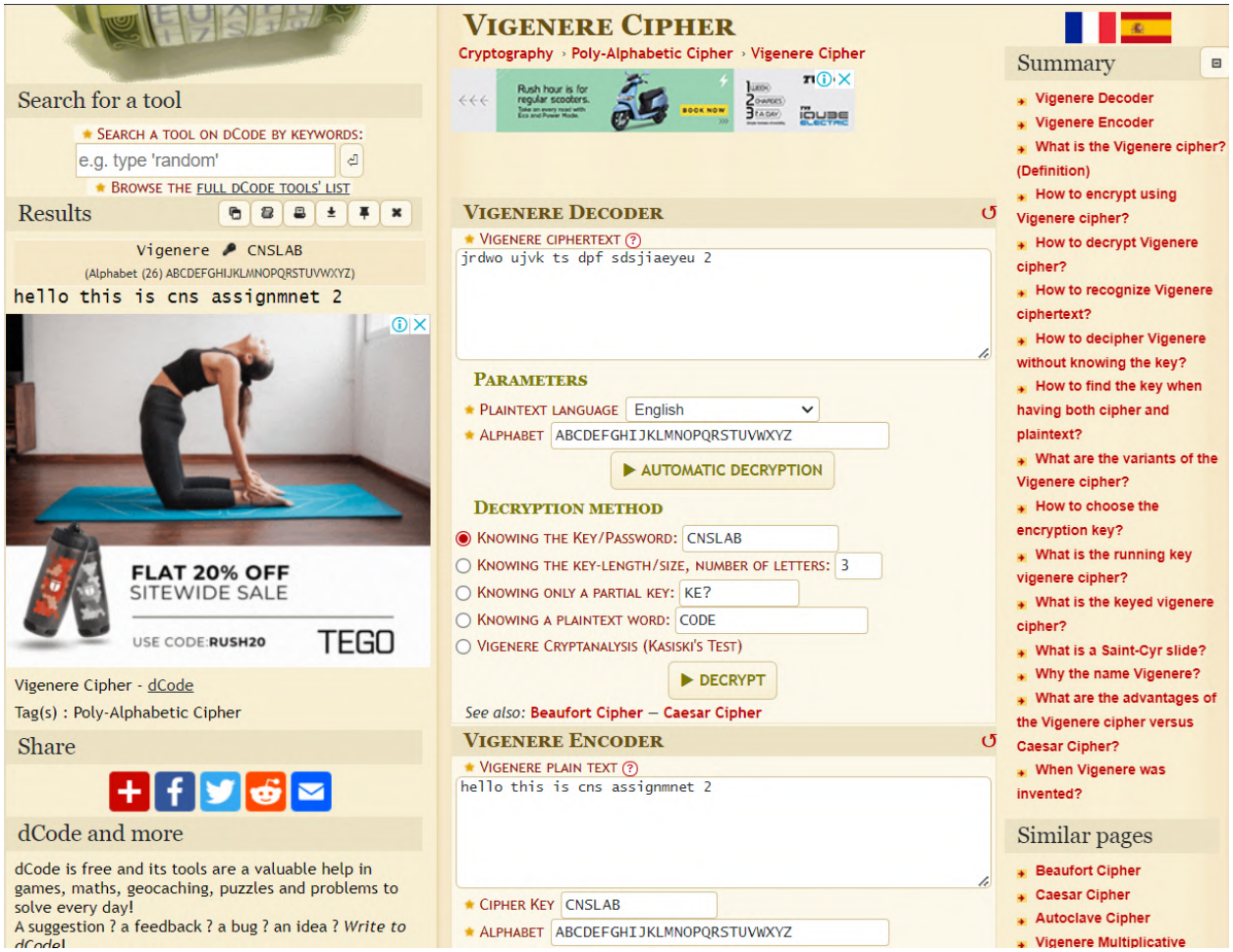
educated guesses about which letters might correspond to common letters in the English language.

Output Screenshot:-

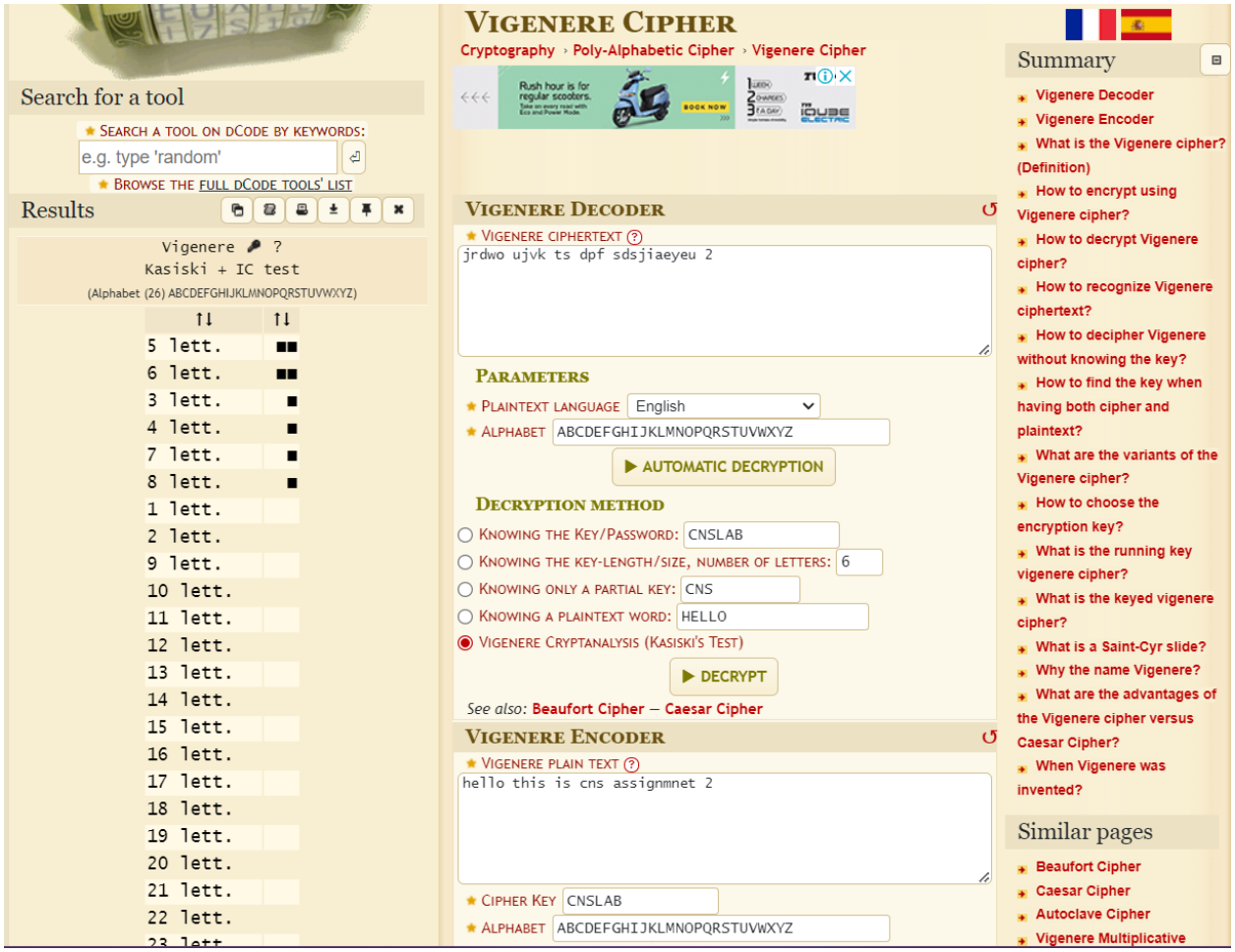
1)Vigenere cipher - encode plain text



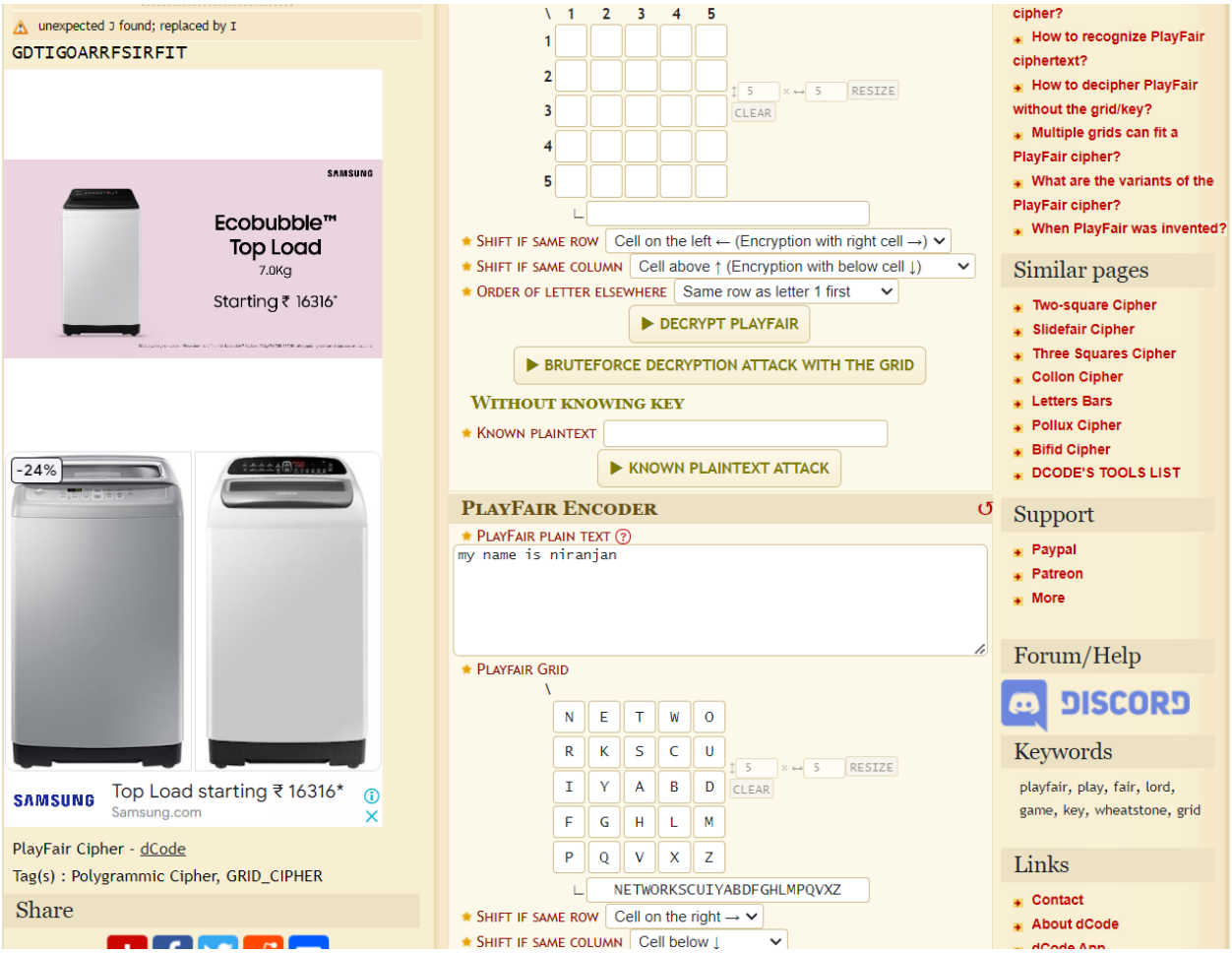
2)Vigenere cipher - decode cipher text



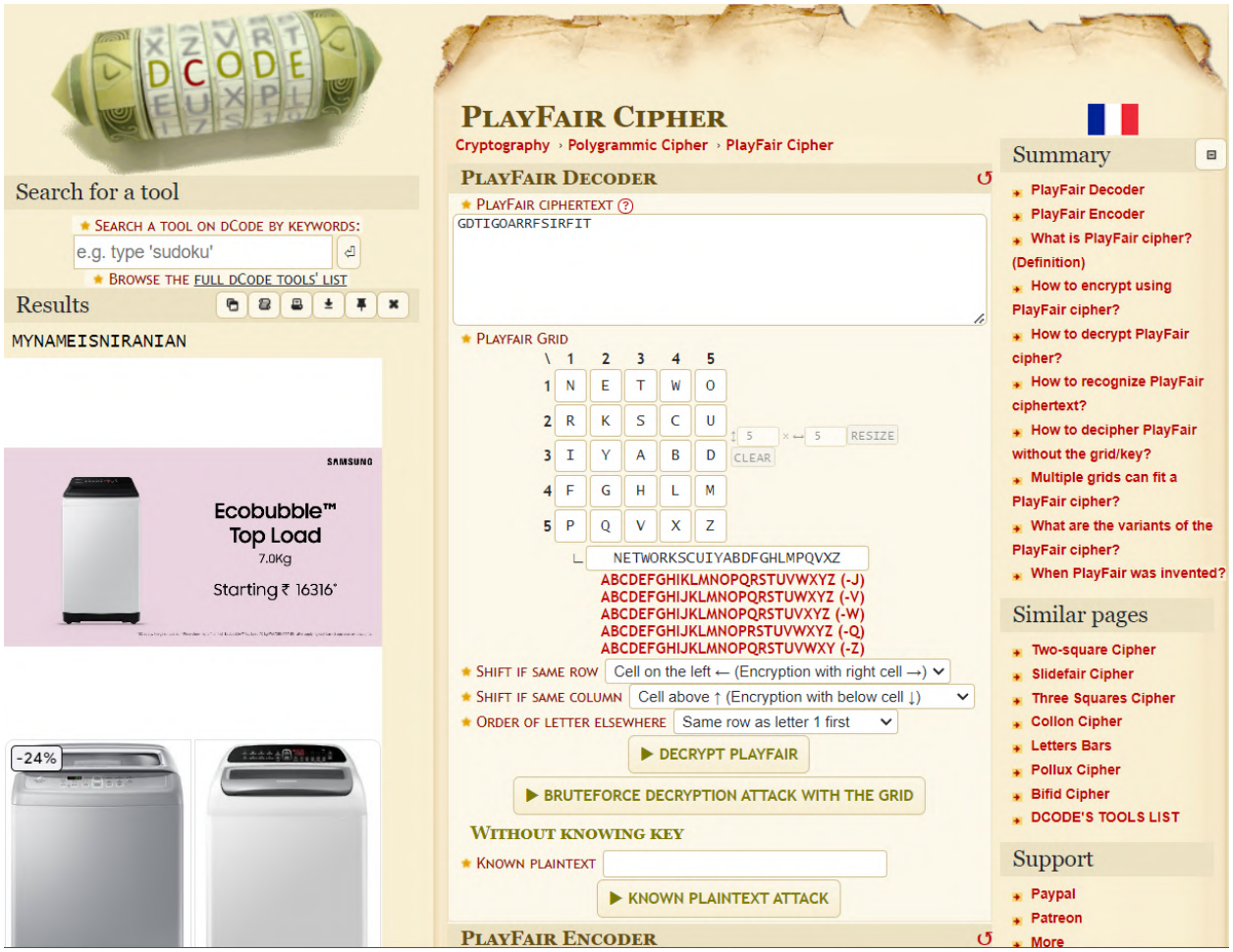
3)Vigenere Cipher - Kasiki test



4)Playfair cipher (key - networksecurity) - encode



5)Playfair cipher - decode



Conclusion:-

Successfully implemented cryptanalysis of vigenere cipher as well as playfair cipher and also encoded plain text , learnt in detail about cryptanalysis ,encoding in vigenere , playfair cipher

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 09/08/2023

Lab Assignment No:-3

Aim:- Block cipher modes of operation using Advanced Encryption Standard(AES)

Lab Outcome Attained :- LO2

Theory:-

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

Advanced Encryption Standard (AES) is a symmetric-key block cipher that is used to encrypt and decrypt data. It is a substitution-permutation network (SPN) cipher, which means that it works by repeatedly substituting and permuting the input data.

Key features of AES:
1)Type of cipher: Symmetric-key block cipher
2)Number of rounds: 10, 12, or 14 rounds, depending on the key size
3)Keysize: 128, 192, or 256 bits
4)Block size: 128 bits
5)Operations in each round:
   Byte substitution (SubBytes)
   Shift rows (ShiftRows)
   Mix columns (MixColumns)
   Add round key (AddRoundKey)

The AES algorithm is considered to be very secure and has been adopted by many organizations and governments around the world. It is used to protect sensitive data in a variety of applications, including:
Email
File encryption
Disk encryption
Wireless networking
Cloud computing

Operations performed in each round of AES:
Byte substitution (SubBytes): This operation replaces each byte in the input block with a new byte that is selected from a lookup table. This table is designed to make the cipher resistant to attacks based on frequency analysis.

Shift rows (ShiftRows): This operation cyclically shifts the rows of the input block by different amounts. This operation makes the cipher resistant to attacks based on differential cryptanalysis.

Mix columns (MixColumns): This operation mixes the columns of the input block using a linear transformation. This operation makes the cipher resistant to attacks based on linear cryptanalysis.

Add round key (AddRoundKey): This operation XORs the input block with the round key. This operation ensures that each round of the cipher depends on the previous rounds, which makes it more difficult to break.

The number of rounds in AES depends on the key size. For a 128-bit key, there are 10 rounds. For a 192-bit key, there are 12 rounds. And for a 256-bit key, there are 14 rounds.
The AES algorithm is a very secure and efficient cipher that is widely used around the world

With diagram explain in brief block cipher modes of operation
1) ECB mode

2) CBC mode
3) OFB mode
4) Counter mode

1)ECB (Electronic Code Book)

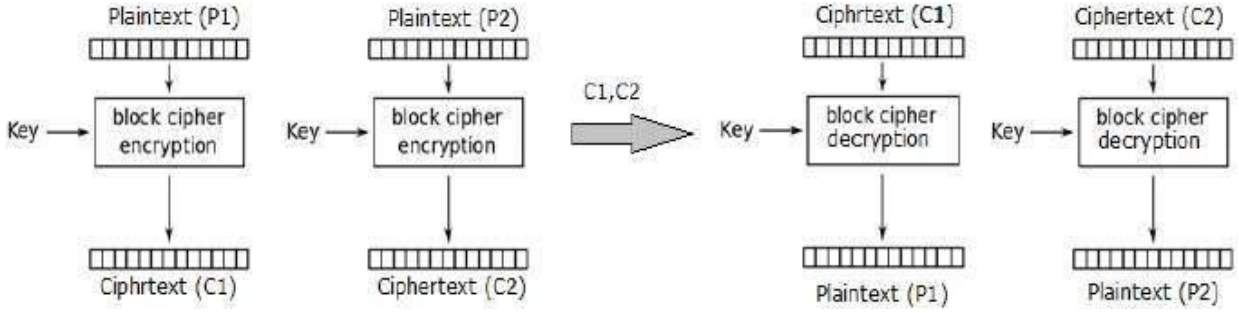
This mode is a most straightforward way of processing a series of sequentially listed message blocks.

Operation

The user takes the first block of plaintext and encrypts it with the key to produce the first block of ciphertext.He then takes the second block of plaintext and follows the same process with same key and so on so forth.

The ECB mode is deterministic, that is, if plaintext block P1, P2,…, Pm are encrypted twice under the same key, the output ciphertext blocks will be the same.

In fact, for a given key technically we can create a codebook of ciphertexts for all possible plaintext blocks. Encryption would then entail only looking up for required plaintext and select the corresponding ciphertext. Thus, the operation is analogous to the assignment of code words in a codebook, and hence gets an official name − Electronic Codebook mode of operation (ECB). It is illustrated as follows −



2)CBC mode(cipher block chaining)

CBC mode of operation provides message dependence for generating ciphertext and makes the system non-deterministic.

Operation

The operation of CBC mode is depicted in the following illustration. The steps are as follows −

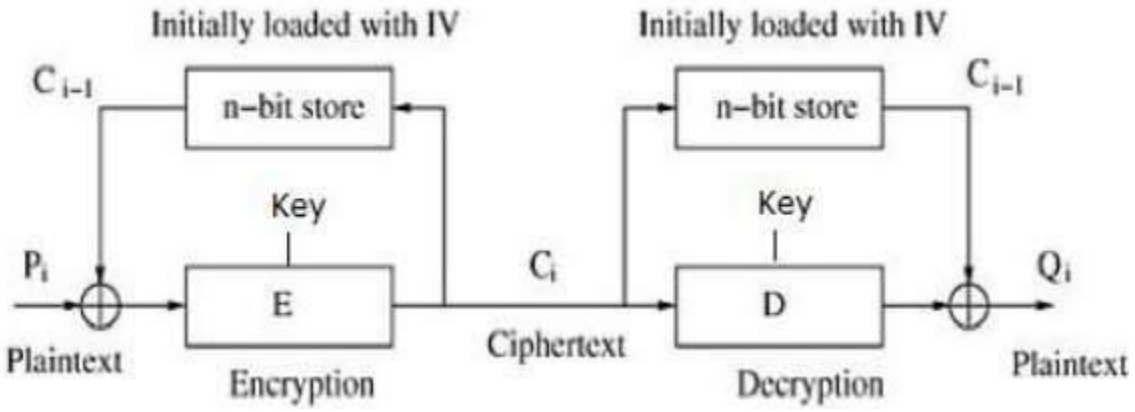
Load the n-bit Initialization Vector (IV) in the top register.

XOR the n-bit plaintext block with data value in top register.

Encrypt the result of XOR operation with underlying block cipher with key K.

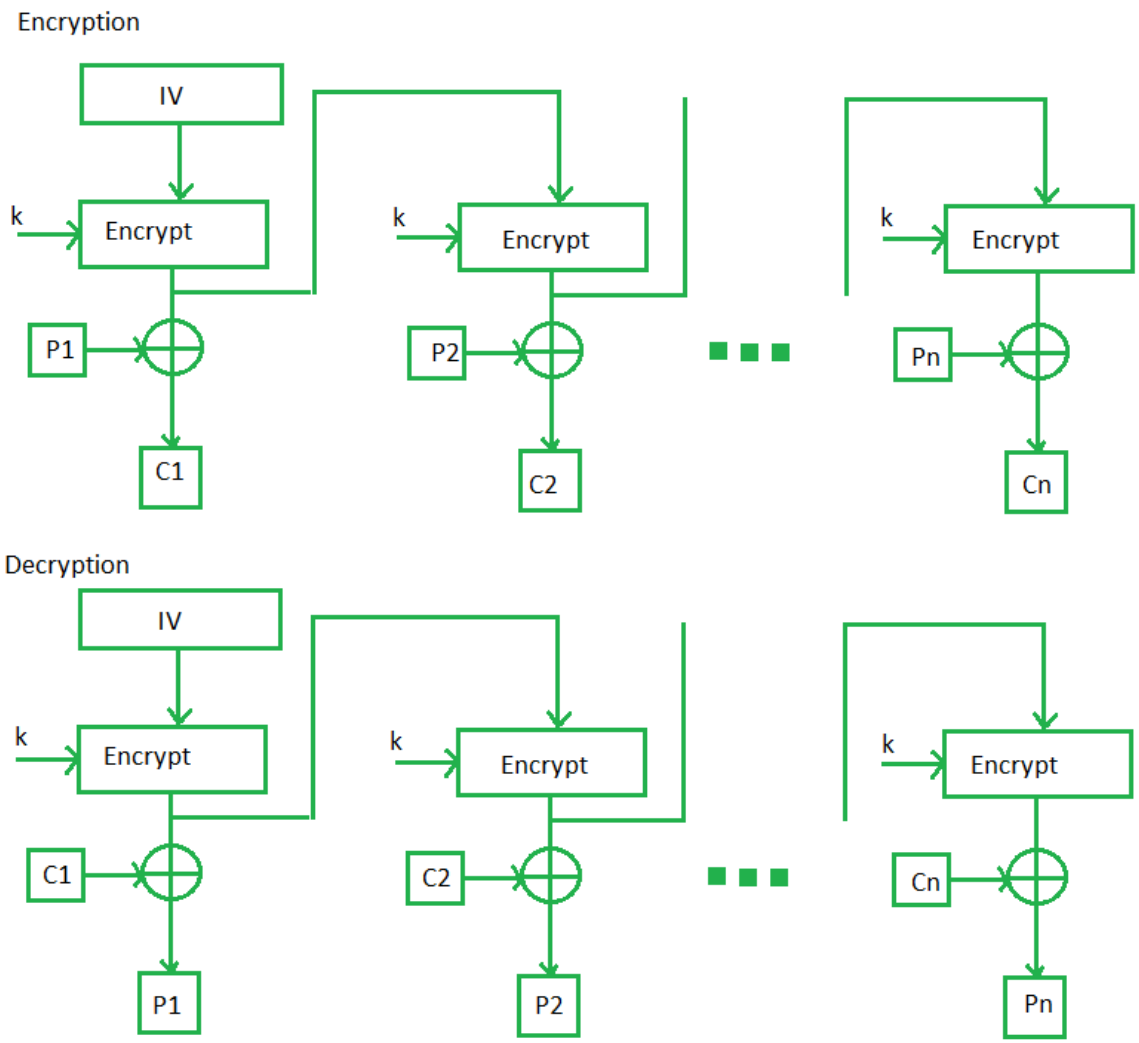
Feed ciphertext block into top register and continue the operation till all plaintext blocks are processed.

For decryption, IV data is XORed with first ciphertext block decrypted. The first ciphertext block is also fed into to register replacing IV for decrypting next ciphertext block.



3)OFB mode(output feedback mode)

OFB Mode stands for output feedback Mode. OFB mode is similar to CFB mode; the only difference is in CFB, the ciphertext is used for the next stage of the encryption process, whereas in OFB, the output of the IV encryption is used for the next stage of the encryption process.The IV is encrypted using the key and form encrypted IV. Plain text and leftmost 8 bits of encrypted IV are combined using XOR and produce the ciphertext.For the next stage, the ciphertext, which is the form in the previous stage, is used as an IV for the next iteration. The same procedure is followed for all blocks.



4)Counter(CTR) mode

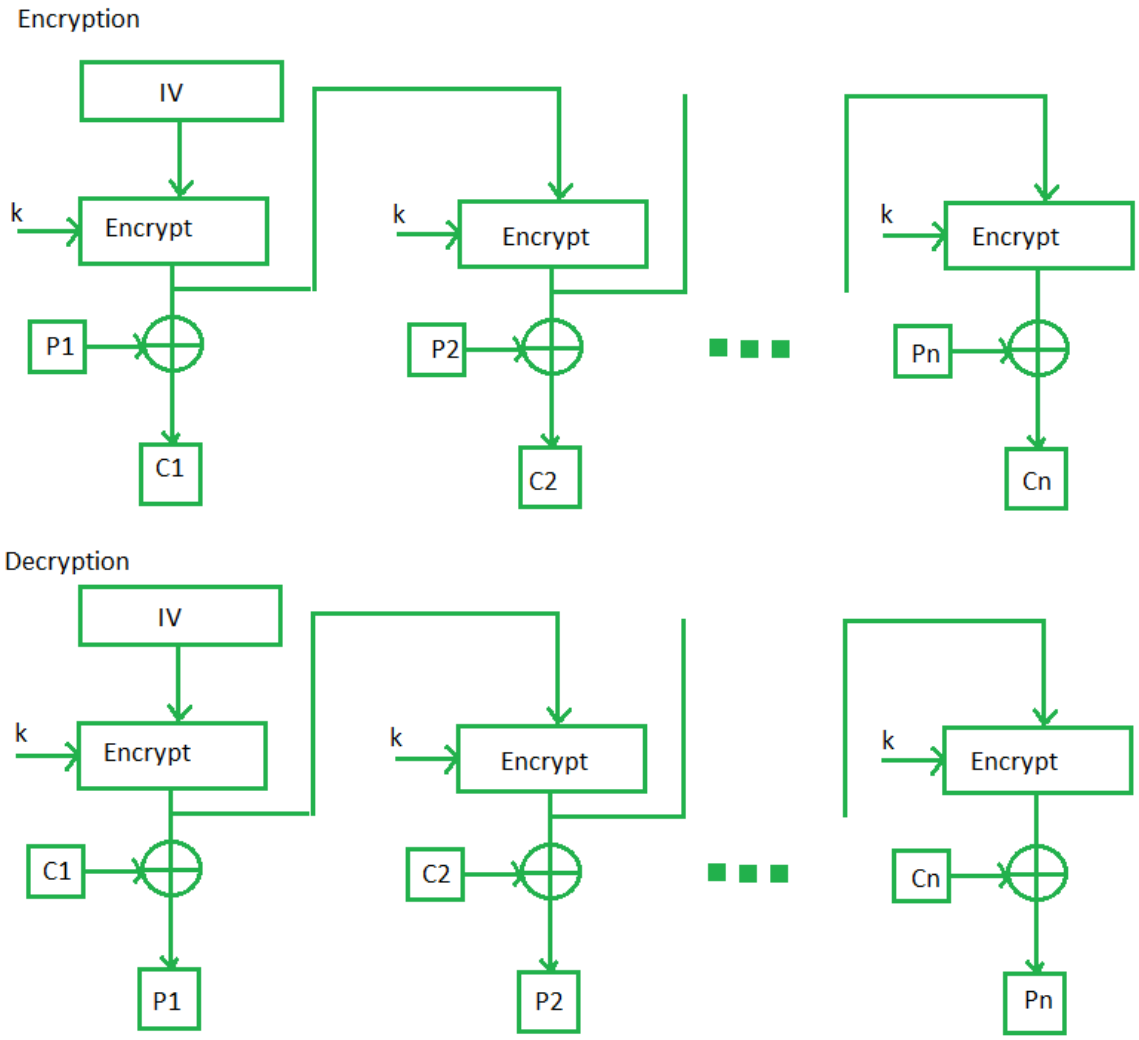
It can be considered as a counter-based version of CFB mode without the feedback. In this mode, both the sender and receiver need to access to a reliable counter, which computes a new shared value each time a ciphertext block is exchanged. This shared counter is not necessarily a secret value, but challenge is that both sides must keep the counter synchronized.

Operation

Both encryption and decryption in CTR mode are depicted in the following illustration. Steps in operation are −

Load the initial counter value in the top register is the same for both the sender and the receiver. It plays the same role as the IV in CFB (and CBC) mode.Encrypt the contents of the counter with the key and place the result in the bottom register.

Take the first plaintext block P1 and XOR this to the contents of the bottom register. The result of this is C1. Send C1 to the receiver and update the counter. The counter update replaces the ciphertext feedback in CFB mode.Continue in this manner until the last plaintext block has been encrypted.The decryption is the reverse process. The ciphertext block is XORed with the output of encrypted contents of counter value. After decryption of each ciphertext block counter is updated as in case of encryption.



Output Screenshot:-

Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

AES and Modes of Operation

Plaintext: 9e02b6c4 6dad8409 a3dc592c 5f49e9c9
5ae4a86a 65c15647 f2b74f22 47dab354
21e25393 4b0a087d 36f79572 f70e32b8
5efe9e6d dd24c2ed 7c941112 9c521b47
b1be277f 63340766 2818260b 135894a9

Next Plaintext

Key: 9d8c0789 a9a3fede 99b87128 a85c7ee1

Next Keytext

IV:

Next IV

CTR:

Next CTR

PART III

Calculate XOR:

Calculate XOR

XOR:

PART IV

Key in hex: 9d8c0789 a9a3fede 99b87128 a85c7ee1

Plaintext in hex: b1be277f 63340766 2818260b 135894a9

Ciphertext in hex: 44b4ae8b c72b19ac 9f56206a aa0cbe4d

Encrypt | Decrypt | Clear

PART V

Enter your answer here:

41b6274c 14cc53f1 6f7af601 c9293182 f742b018 52d5ede3 4397270d 80c21

Check Answer!

CORRECT!!

Type here to search | ENG | 14:25 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

AES and Modes of Operation

PART I

Choose your mode of operation: Output Feedback

PART II

Key size in bits: 128

Plaintext: efbfd9b5 16be4bf5 3f4a32ae 18225641
a28e6b05 f9dd6d0e 2ceb4ac6 43e0bcb0
4f4fda79 65b7567c bede510c 3feceea7
5b7befac 25904cc9 e8246988 e1c02e51
4f6a92c1 6607fca4 a1682d56 fbf0b537

Next Plaintext

Key: 969827e3 18d136da cce9794a 9fe9911c

Next Keytext

IV: d7d68add bc0a6bad 4b16082b 8a62c28a

Next IV

PART III

Calculate XOR:

4f6a92c1 6607fca4 a1682d56 fbf0b537

1f6b8715 33427730 88c30c37 954c1685

Calculate XOR

XOR: 500115d4 55458b94 29ab2161 6ebca3b2

PART IV

Key in hex: 969827e3 18d136da cce9794a 9fe9911c

Plaintext in hex: 0183e3a3 5b614f98 eac112d1 16daea81

Ciphertext in hex: 1f6b8715 33427730 88c30c37 954c1685

Encrypt | Decrypt | Clear

Type here to search | ENG | 14:56 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

AES and Modes of Operation

Key size in bits: 128

Plaintext:
```
efbfd9b5 16be4bf5 3f4a32ae 18225641
a28e6b05 f9dd6d0e 2ceb4ac6 43e0bcb0
4f4fda79 65b7567c bede510c 3feceea7
5b7befac 25904cc9 e8246988 e1c02e51
4f6a92c1 6607fca4 a1682d56 fbf0b537
```
Next Plaintext

Key: 969827e3 18d136da cce9794a 9fe9911c

Next Keytext

IV: d7d68add bc0a6bad 4b16082b 8a62c28a

Next IV

PART III

Calculate XOR:

4f6a92c1 6607fca4 a1682d56 fbf0b537

1f6b8715 33427730 88c30c37 954c1685

Calculate XOR

XOR: 500115d4 55458b94 29ab2161 6ebca3b2

PART IV

Key in hex: 969827e3 18d136da cce9794a 9fe9911c

Plaintext in hex: 0183e3a3 5b614f98 eac112d1 16daea81

Ciphertext in hex: 1f6b8715 33427730 88c30c37 954c1685

Encrypt | Decrypt | Clear

PART V

Enter your answer here:

d7d68add bc0a6bad 4b16082b 8a62c28a 74fb98c8 11c9008d c0dedba0 a29

Check Answer!

CORRECT!!

Type here to search | ENG | 14:56 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

AES and Modes of Operation

PART I

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

PART II

Key size in bits: 128

Plaintext:
```
9e02b6c4 6dad8409 a3dc592c 5f49e9c9
5ae4a86a 65c15647 f2b74f22 47dab354
21e25393 4b0a087d 36f79572 f70e32b8
5efe9e6d dd24c2ed 7c941112 9c521b47
b1be277f 63340766 2818260b 135894a9
```
Next Plaintext

Key: 9d8c0789 a9a3fede 99b87128 a85c7ee1

Next Keytext

IV:

Next IV

CTR:

Next CTR

PART III

Calculate XOR:

Calculate XOR

XOR:

PART IV

Key in hex: 9d8c0789 a9a3fede 99b87128 a85c7ee1

Plaintext in hex: b1be277f 63340766 2818260b 135894a9

Ciphertext in hex: 44b4ae8b c72b19ac 9f56206a aa0cbe4d

Encrypt | Decrypt | Clear

Type here to search | ENG | 15:01 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

AES and Modes of Operation

Plaintext:
```
9e02b6c4 6dad8409 a3dc592c 5f49e9c9
5ae4a86a 65c15647 f2b74f22 47dab354
21e25393 4b0a087d 36f79572 f70e32b8
5efe9e6d dd24c2ed 7c941112 9c521b47
b1be277f 63340766 2818260b 135894a9
```
Next Plaintext

Key: 9d8c0789 a9a3fede 99b87128 a85c7ee1 Next Keytext

IV: Next IV

CTR: Next CTR

PART III

Calculate XOR:

Calculate XOR

XOR:

PART IV

Key in hex: 9d8c0789 a9a3fede 99b87128 a85c7ee1

Plaintext in hex: b1be277f 63340766 2818260b 135894a9

Ciphertext in hex: 44b4ae8b c72b19ac 9f56206a aa0cbe4d

Encrypt Decrypt Clear

PART V

Enter your answer here:

41b6274c 14cc53f1 6f7af601 c9293182 f742b018 52d5ede3 4397270d 80c21 Check Answer!

CORRECT!!

Type here to search ENG 15:01 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

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

Plaintext:
```
c096db76 bc084d51 a0dc9fe9 b3e2f4b8
5eed2064 68029863 59b71c0c b86e91c1
66e3a8fd 4a183dc8 d2b75f18 dc305e0f
8c03d450 12880f54 03469256 ab884d88
67c2648a e98d960b 7e0110ac e8e31045
```
Next Plaintext

Key: 9c9fe223 03d2fbe2 88c441e5 0b58ed7d Next Keytext

IV: e747d16b c355ccff c80ae504 06a3e645 Next IV

PART III

Calculate XOR:

67c2648a e98d960b 7e0110ac e8e31045

728527d5 c5d3ef1e 14561029 310f1652 Calculate XOR

XOR: 1547435f 2c5e7915 6a570085 d9ec0617

PART IV

Key in hex: 9c9fe223 03d2fbe2 88c441e5 0b58ed7d

Plaintext in hex: 1547435f 2c5e7915 6a570085 d9ec0617

Ciphertext in hex: 85c0eed1 06502ed7 7b1e1877 9c441b3c

Encrypt Decrypt Clear

Type here to search ENG 15:47 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

Virtual Labs

AES and Modes of Operation

Key size in bits: 128

Plaintext:
```
c096db76 bc084d51 a0dc9fe9 b3e2f4b8
5eed2064 68029863 59b71c0c b06e91c1
66e3a8fd 4a183dc8 d2b75f18 dc305e0f
8c03d450 12880f54 03469256 ab884d88
67c2648a e98d960b 7e0110ac e8e31045
```
Next Plaintext

Key: 9c9fe223 03d2fbe2 88c441e5 0b58ed7d

Next Keytext

IV: e747d16b c355ccff c80ae504 06a3e645

Next IV

PART III

Calculate XOR:

67c2648a e98d960b 7e0110ac e8e31045

728527d5 c5d3ef1e 14561029 310f1652

Calculate XOR

XOR: 1547435f 2c5e7915 6a570085 d9ec0617

PART IV

Key in hex: 9c9fe223 03d2fbe2 88c441e5 0b58ed7d

Plaintext in hex: 1547435f 2c5e7915 6a570085 d9ec0617

Ciphertext in hex: 85c0eed1 06502ed7 7b1e1877 9c441b3c

Encrypt | Decrypt | Clear

PART V

Enter your answer here:

e747d16b c355ccff c80ae504 06a3e645 1e3ad250 9e6d7584 99966612 5927

Check Answer!

CORRECT!!

Type here to search | ENG | 15:47 28-08-2023

Virtual Labs | Virtual Labs | Virtual Labs | Virtual Labs

cse29-iiith.vlabs.ac.in/exp/aes/simulation.html

Page Unresponsive

You can wait for it to become responsive or exit the page.

Virtual Labs

Wait | Exit page

Key size in bits: 128

Plaintext:
```
809c1256 57bb822e 16793620 b1f6cb74
f418da4f e126b410 82e74c6d d75cfb58
b2826fe5 1713520e b1c3006f 5b796bcb
cedde644 185e29b5 5ff1e8a3 3454b701
103b695d ac55a91a 5981ea82 d4681731
```
Next Plaintext

Next Keytext

CTR: f24d9cb5 e987b0d9 56d7d23e d043426e

Next CTR

PART III

Calculate XOR:

103b695d ac55a91a 5981ea82 d4681731

7db24f44 1a37f06d 31dfa2e5 5f989225

Calculate XOR

XOR: 6d892619 b6625977 685e4867 8bf08514

PART IV

Key in hex: 2967c5fd 926fa06d 9c87ab27 8890f660

Plaintext in hex: f24d9cb5 e987b0d9 56d7d23e d043426e

Ciphertext in hex: 7db24f44 1a37f06d 31dfa2e5 5f989225

Encrypt | Decrypt | Clear

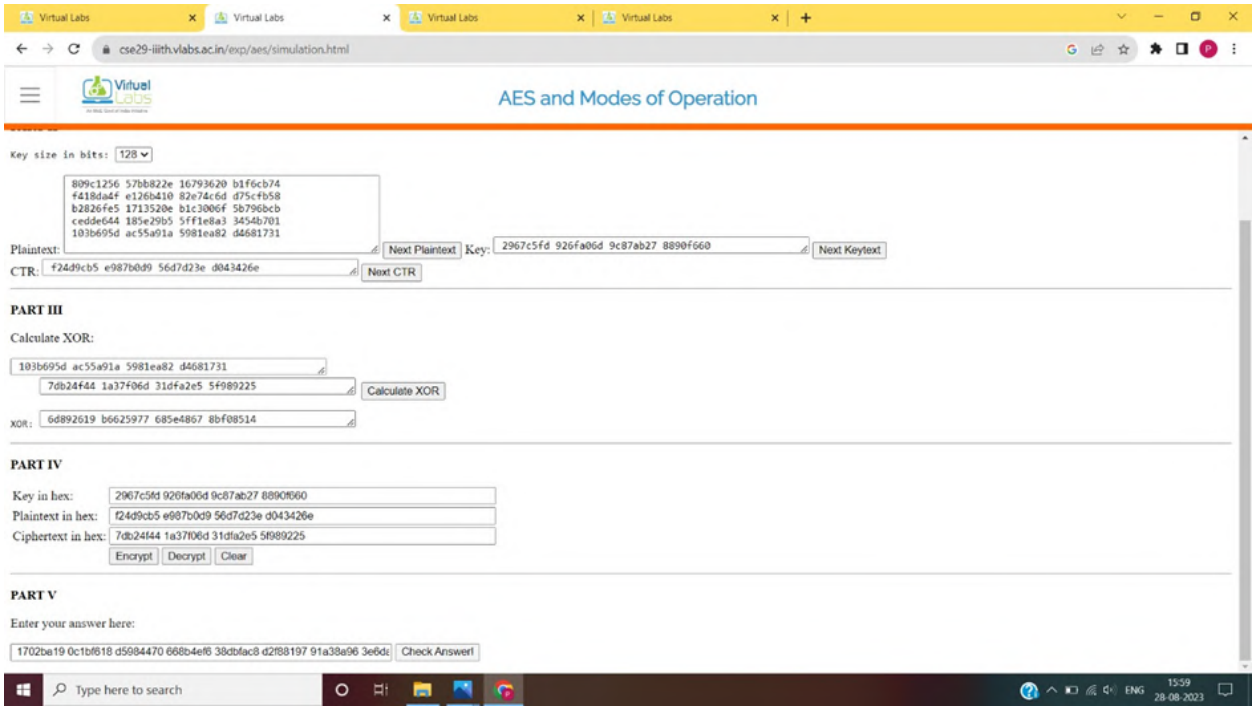
PART V

Enter your answer here:

1702ba19 0c1bf618 d5984470 668b4ef6 38dbfac8 d2f88197 91a38a96 3e6d

Check Answer!

Type here to search | ENG | 15:59 28-08-2023



Conclusion:- Learnt about various modes of operation in block cipher such as ECB , CBC , OFB , CTR mode and also practically implemented them for randomly generated strings and performed AES on that string to get encrypted message

Roll Number:- 2205051
Name:-Niranjan Joshi
Date:- 26/07/2023

Lab Assignment No:-4

Aim:- Implementation and analysis of RSA cryptosystem and digital signature scheme using RSA

Lab Outcome Attained :- LO2

Theory:-

Explain the steps of RSA key generation

Steps involved in RSA key generation are as follows:-

1)Generate two large prime numbers, p and q.

2)Calculate the modulus n as n = p * q.

3)Calculate the totient $\phi(n)$ as $\phi(n) = (p - 1)(q - 1)$.

4)Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(\phi(n), e) = 1$.

5)Calculate the private exponent d as $d = e^{(-1)} \bmod \phi(n)$.

6)The public key is the pair (n, e), and the private key is the pair (n, d).

Eg:-

Let $p = 7$ and $q = 11$. Then, $n = p * q = 77$. The totient $\phi(n)$ is $(p - 1)(q - 1) = 6 * 10 = 60$. An integer e that satisfies the condition $1 < e < \phi(n)$ and $\gcd(\phi(n), e) = 1$ is $e = 5$. The private exponent d is calculated as $d = e^{(-1)} \bmod \phi(n) = 3$.

The public key is the pair $(n, e) = (77, 5)$. The private key is the pair $(n, d) = (77, 3)$.

In this example, the public key is (77, 5), and the private key is (77, 3). These keys can be used to encrypt and decrypt messages.

To encrypt a message, the sender would use the public key to encrypt the message. The receiver would then use the private key to decrypt the message.

For example, to encrypt the message "hello" using the public key (77, 5), the sender would first convert the message to a number. In this case, the number would be 104. The sender would then encrypt the number using the public key. The encrypted number would be 525.

The receiver would then use the private key (77, 3) to decrypt the number. The decrypted number would be 104. The receiver would then convert the number back to the message, which would be "hello".

Explain the steps of digital signature verification and generation process .

steps in generating and verifying a digital signature :

1. Generate the public and private keys.

The public and private keys are generated using the RSA key generation algorithm. The public key is made public, while the private key is kept secret.

2. Create a hash of the message.

The message to be signed is hashed using a cryptographic hash function. The hash function produces a fixed-length value, called the hash digest, that uniquely represents the message.

3. Encrypt the hash digest with the private key.

The hash digest is encrypted with the private key. The encrypted hash digest is the digital signature.

4. Send the message and the digital signature to the recipient.

The sender sends the message and the digital signature to the recipient.

5. Verify the digital signature.

The recipient decrypts the digital signature with the public key. The decrypted hash digest is compared to the hash of the message. If the two hashes match, then the signature is valid.

overview of the steps involved in generating and verifying a digital signature:

Generation

Generate the public and private keys.

Create a hash of the message.

Encrypt the hash digest with the private key.

Verification

Decrypt the digital signature with the public key.

Compare the decrypted hash digest to the hash of the message.

Output Screenshot:-

1) RSA key generation and decode

Plaintext (string):

hello this is implementation of rsa algorithm

encrypt

Ciphertext (hex):

```
469f99c7c7bd6eb33fb07b16aa06cbdbb8dc4ba02bcc9abb46de3ba1e5110ed4
3a6851ca040fb6114e8b8e2306dd1b55d7fcf2d89990b5f62eee8a9806b8f263
```

decrypt

Decrypted Plaintext (string):

hello this is implementation of rsa algorithm

Status:

Decryption Time: 5ms

RSA private key

1024 bit | 1024 bit (e=3) | 512 bit | 512 bit (e=3) | Generate | bits = 512

Modulus (hex):

```
BC86E3DC782C446EE756B874ACECF2A115E613021EAF1ED5EF295BEC2BED899D
26FE2EC896BF9DE84FE381AF67A7B7CBB48D85235E72AB595ABF8FE840D5F8DB
```

Public exponent (hex, F4=0x10001):

3

Private exponent (hex):

```
7daf4292fac82d9f44e47af87348a1c0b9440cac1474bf394a1b929d729e5bbc
f402f29a9300e11b478c091f7e5dacd3f8edae2effe3164d7e0eeada87ee817b
```

P (hex):

```
ef3fc61e21867a900e01ee4b1ba69f5403274ed27656da03ed88d7902cce693f
```

Q (hex):

```
c9b9fcc298b7d1af568f85b50e749539bc01b10a68472fe1302058104821cd65
```

D mod (P-1) (hex):

```
9f7fd9696baefc6009569edcbd19bf8d576f89e1a439e6ad4905e50ac8899b7f
```

D mod (Q-1) (hex):

```
867bfdd7107a8bca39b503ce09a30e267d567606f02f7540cac03ab5856bde43
```

1/Q mod P (hex):

```
412d6b551d93ee1bd7dccafc63d7a6d031fc66035ecc630ddf75f949a378cd9d
```

2)Digital Signature (hash key generation)

Digital Signatures Scheme

Digitally sign the plaintext with Hashed RSA.

Plaintext (string):

hello this is generation of digital signature SHA-1

Hash output(hex):

5d41539793f25323c88168be360b2969d153514b

3)Digital Signature(generation)

Input to RSA(hex):

5d41539793f25323c88168be360b2969d153514b | Apply RSA

Digital Signature(hex):

```
5ce2601899d9c6fcdc5173556d9a8615d4fb021dabdde2337c2fbb4750bcf2f2
5d8f38e2a3e865a91b554ce7237aabb11b322760febc5ba53dfb7cbb6dee1632
531c478a491963bb50fb98328eb198f4a4cf54232cb1683424fa38d1f879e177
dae72e12b5c9cb41556644febdec9eba0d04eb79d1cd87540cf7b02893aa62b0
```

Digital Signature(base64):


```
XOJgGJnZxvzcUXNVbZqGFdT7Ah2r3eIzfC+7R1C88vJdjzjio+hlqRtVTOcjequx
GzInYP68W6U9+3y7be4WMlMcR4pJGWO7UPuYMo6xmPSkz1QjLLFoNCT6ONH4eeF3
2ucuErXJy0FVZkT+veyeug0E63nRzYdUDPewKJOqYrA=
```


Status:

Time: 2ms

RSA public key

Public exponent (hex, F4=0x10001):

3

Modulus (hex):

```
ABC30681295774F7CECA691EC17F4E762DA6DE70F198EAEE3CCE3A435FC006B9
71DC24E55904F1D2705758C041C2B0B18E8BFAE2C9CD96B50082D7D8C7342CBA
B7F6E0622DA53B8B56DBDB24174F00173263CFECAE604795CDA2A037BC3A69B7
C0090AA2DE1568998BCD6D70CC2E0574755B9F7986AE01CE8714A26144279CDB
```

1024 bit | 1024 bit (e=3) | 512 bit | 512 bit (e=3)

Conclusion:-

Learnt about RSA scheme and RSA cryptosystem , explored steps involved in digital key generation and verification , generated and verified digital signature using software and also implemented RSA scheme

RollNumber:2105051
Name:-Niranjan Joshi
Date:- 09/08/2023

Lab Assignment No:-5

Aim:

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

Lab Outcome Attained :- LO2

Theory:-

What is need of hashing ? List different hashing algorithms .

Hashing is a fundamental concept in computer science and cryptography. It involves the transformation of input data (such as a file, password, or message) into a fixed-size value or hash code, typically represented as a sequence of characters or numbers. The primary purposes of hashing are:

Data Integrity: Hashing is used to ensure data integrity by generating a unique hash value for a piece of data. If the data changes, even by a single bit, the resulting hash will be completely different. This property is useful for detecting data corruption or tampering.

Data Retrieval: Hashing is used in data structures like hash tables to quickly retrieve values based on their keys. Hash functions allow for efficient storage and retrieval of data, reducing lookup times.

Password Storage: Hashing is crucial for securely storing passwords. Instead of storing actual passwords, systems store the hash of passwords. When a user logs in, their entered password is hashed and compared to the stored hash. This adds an extra layer of security because the original password is not stored in plaintext.

Digital Signatures: Hashing is used to create digital signatures, which are cryptographic mechanisms that provide authenticity and integrity to digital documents, messages, or transactions.

Cryptographic Applications: Hashing is used in cryptographic algorithms to create secure message digests, key derivation functions, and more.

Different Hashing Algorithms:
MD5 (Message Digest Algorithm 5): MD5 produces a 128-bit hash value. However, MD5 is considered weak and insecure due to vulnerabilities that allow for collision attacks.

SHA-1 (Secure Hash Algorithm 1): SHA-1 produces a 160-bit hash value. Like MD5, SHA-1 is also considered weak and has been deprecated in favor of more secure algorithms.

SHA-256, SHA-384, SHA-512: These are part of the SHA-2 family and produce hash values of different lengths (256, 384, and 512 bits, respectively). They are widely used and considered secure for many cryptographic applications.

SHA-3: The latest member of the Secure Hash Algorithm family, SHA-3 produces hash values of various lengths and is designed to be highly secure and resistant to known attacks.

bcrypt: A key derivation function commonly used for securely hashing passwords. It incorporates a work factor that slows down hashing and makes brute-force attacks more difficult.

Argon2: A state-of-the-art key derivation function that won the Password Hashing Competition. It's designed to be memory-hard and resistant to various types of attacks.

HMAC (Hash-based Message Authentication Code): Not a standalone hash algorithm, but a construction that uses a hash function in combination with a secret key to provide message integrity and authenticity.

BLAKE2: A high-speed cryptographic hash function that is an improvement over SHA-2 and SHA-3 in terms of performance.

Write the commands used for generating hash values , matching them with stored hash value and auditing using hashdeep tool .

hashdeep is a command-line tool used for generating hash values, matching them with stored hash values, and performing audits for files and directories. It's a versatile tool that supports multiple hash algorithms and provides a way to verify data integrity and authenticity.

Basic commands for generating hash values, matching them with stored hash values, and performing audits using hashdeep:

Generating Hash Values:
To generate hash values for files in a directory and its subdirectories, command:

hashdeep -r -l -c SHA256 /path/to/directory > hash_values.txt
This command **generates SHA-256 hash values for all files in the specified directory** and its subdirectories and saves the hash values along with file paths to the hash_values.txt file.

Matching Hash Values:
To match hash values with stored hash values,command:

hashdeep -r -l -c -a -k hash_values.txt /path/to/directory
This command compares the hash values in the hash_values.txt file with the hash values of files in the specified directory and its subdirectories. It will report any discrepancies or mismatches.

Auditing Files:
To perform an audit of files based on hash values, command:

hashdeep -r -l -c -k -x audit_results.txt /path/to/directory
This command compares the hash values of files in the specified directory and its subdirectories with the hash values stored in the hash_values.txt file. It will generate an audit report and save it to the audit_results.txt file.

Output Screenshots :

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:47.945571 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.945655 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.945770 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.945907 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.945958 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946091 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946300 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.946355 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946396 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946485 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946568 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946642 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946738 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946836 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946891 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.946991 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947023 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947086 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947165 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947217 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947253 IP 192.168.0.115.55819 > 142.250.192.67.443: UDP, length 40
12:15:47.947262 IP 192.168.0.115.55819 > 142.250.192.67.443: UDP, length 40
12:15:47.947293 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947352 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947401 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947594 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.947636 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947758 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947790 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947850 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.947883 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948005 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.948055 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948164 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948392 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948490 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948558 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.948684 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:47.933724 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.933908 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 646
12:15:47.934016 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.934177 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 631
12:15:47.935071 IP 192.168.0.115.55819 > 142.250.192.67.443: UDP, length 40
12:15:47.936172 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.936289 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.936334 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.936458 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.936584 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.936623 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.936657 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.936696 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.936880 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.936921 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937027 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937163 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937274 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937313 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937444 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.937497 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.938687 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.938782 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.938844 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.938893 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.939043 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.939086 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.939118 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.941581 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.942077 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942248 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942332 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942390 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942461 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942582 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942714 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942819 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.942954 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:45.640598 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 31
12:15:45.645597 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 593
12:15:45.668750 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 32
12:15:45.711544 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 36
12:15:45.712774 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 31
12:15:45.721876 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 268
12:15:45.748120 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 32
12:15:45.786087 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 36
12:15:45.786966 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 31
12:15:45.794303 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 268
12:15:45.817702 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 32
12:15:45.860009 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 36
12:15:45.862126 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 31
12:15:47.021572 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 690
12:15:47.044331 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 34
12:15:47.090515 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.261108 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 745
12:15:47.284732 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 34
12:15:47.356881 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:47.363309 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 652
12:15:47.389802 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 35
12:15:47.860039 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 40
12:15:47.884646 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 814
12:15:47.907213 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:47.928793 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 653
12:15:47.929061 IP 192.168.0.115.35858 > 192.168.0.1.53: 64156+ [1au] A? ssl.gstatic.com. (44)
12:15:47.929125 IP 192.168.0.115.46878 > 192.168.0.1.53: 33271+ [1au] AAAA? ssl.gstatic.com. (44)
12:15:47.929529 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 668
12:15:47.930135 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 666
12:15:47.930557 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 675
12:15:47.931249 IP 192.168.0.115.55819 > 142.250.192.67.443: UDP, length 1357
12:15:47.931379 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:47.931529 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 672
12:15:47.931754 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 670
12:15:47.932011 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 672
12:15:47.932362 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 672
12:15:47.932723 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 672
12:15:47.932981 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:45.394469 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [P.], seq 517:581, ack 4329, win 501, options [nop,nop,TS val 592184513 ecr
 2374550278], length 64
12:15:45.394952 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [P.], seq 581:751, ack 4329, win 501, options [nop,nop,TS val 592184513 ecr
 2374550278], length 170
12:15:45.396203 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [.], ack 4943, win 501, options [nop,nop,TS val 592184515 ecr 2374550288],
length 0
12:15:45.396800 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [P.], seq 751:782, ack 4943, win 501, options [nop,nop,TS val 592184515 ecr
 2374550288], length 31
12:15:45.397729 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [.], ack 4974, win 501, options [nop,nop,TS val 592184516 ecr 2374550289],
length 0
12:15:45.433723 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 36
12:15:45.435367 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 31
12:15:45.437539 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 1357
12:15:45.440658 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.451120 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.451135 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.458585 ARP, Reply 192.168.0.115 is-at 04:0e:3c:1a:5c:ad, length 28
12:15:45.470681 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.470695 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.500177 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 42
12:15:45.505676 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 43
12:15:45.508017 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [P.], seq 853:1280, ack 1404, win 501, options [nop,nop,TS val 1544065109 ecr
 2028888043], length 427: HTTP: POST /gts1c3 HTTP/1.1
12:15:45.509045 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.509061 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.538090 IP 192.168.0.115.33240 > 142.250.66.2.443: Flags [S], seq 1421234689, win 64240, options [mss 1460,sackOK,TS val 3205774662 ec
r 0,nop,wscale 7], length 0
12:15:45.540579 IP 192.168.0.115.33240 > 142.250.66.2.443: Flags [.], ack 3517905791, win 502, options [nop,nop,TS val 3205774664 ecr 58830299
4], length 0
12:15:45.546769 IP 192.168.0.115.33240 > 142.250.66.2.443: Flags [P.], seq 0:517, ack 1, win 502, options [nop,nop,TS val 3205774670 ecr 58830
2994], length 517
12:15:45.546967 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.547015 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 40
12:15:45.570615 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [.], ack 2106, win 501, options [nop,nop,TS val 1544065171 ecr 2028888255], l
ength 0
12:15:45.573436 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 108
12:15:45.573496 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 66
12:15:45.574401 IP 192.168.0.115.52206 > 142.250.66.2.443: UDP, length 365
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:45.123093 IP 192.168.0.115.44636 > 142.251.42.78.443: Flags [P.], seq 751:782, ack 4956, win 501, options [nop,nop,TS val 1243076793 ecr
 1613452846], length 31
12:15:45.128159 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 43
12:15:45.128586 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.129435 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 44
12:15:45.130954 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.131016 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.133756 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 150
12:15:45.134416 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 673
12:15:45.137241 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 31
12:15:45.165403 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.166469 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 47
12:15:45.166971 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.206190 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 36
12:15:45.206383 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 31
12:15:45.213550 IP 192.168.0.115.50545 > 192.168.0.1.53: 4797+ [1au] A? adservice.google.co.in. (51)
12:15:45.213878 IP 192.168.0.115.37656 > 192.168.0.1.53: 40078+ [1au] AAAA? adservice.google.co.in. (51)
12:15:45.214441 IP 192.168.0.115.56260 > 192.168.0.1.53: 57347+ [1au] A? googleads.g.doubleclick.net. (56)
12:15:45.214717 IP 192.168.0.115.41288 > 192.168.0.1.53: 30603+ [1au] AAAA? googleads.g.doubleclick.net. (56)
12:15:45.218936 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 1357
12:15:45.223530 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.237199 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.237248 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.263894 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.263950 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.282073 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 42
12:15:45.289970 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 43
12:15:45.290423 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 45
12:15:45.290463 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.296169 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [P.], seq 426:853, ack 702, win 501, options [nop,nop,TS val 1544064897 ecr 2
028887614], length 427: HTTP: POST /gts1c3 HTTP/1.1
12:15:45.313086 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 621
12:15:45.315102 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [S], seq 1523800880, win 64240, options [mss 1460,sackOK,TS val 592184434 e
cr 0,nop,wscale 7], length 0
12:15:45.315776 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.316284 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.316339 IP 192.168.0.115.33359 > 142.250.66.2.443: UDP, length 40
12:15:45.317511 IP 192.168.0.115.37330 > 172.217.166.66.443: Flags [.], ack 903489962, win 502, options [nop,nop,TS val 592184436 ecr 23745502
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:44.994207 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 671
12:15:44.996184 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.997466 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:44.998557 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 683
12:15:45.001967 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 738
12:15:45.004126 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.027501 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 36
12:15:45.027686 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.032830 IP 192.168.0.115.44636 > 142.251.42.78.443: Flags [.], ack 4311, win 485, options [nop,nop,TS val 1243076702 ecr 1613452755],
length 0
12:15:45.040159 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.043928 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 896
12:15:45.056494 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [P.], seq 1278:1704, ack 2104, win 501, options [nop,nop,TS val 1544064657 ec
r 187502018], length 426: HTTP: POST /gts1c3 HTTP/1.1
12:15:45.057711 IP 192.168.0.115.55273 > 192.168.0.1.53: 52970+ [1au] A? adservice.google.com. (49)
12:15:45.057865 IP 192.168.0.115.43746 > 192.168.0.1.53: 12539+ [1au] AAAA? adservice.google.com. (49)
12:15:45.059139 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 685
12:15:45.060676 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 1357
12:15:45.060946 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.064593 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.066104 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.067417 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.074865 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.074882 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.084393 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 31
12:15:45.084467 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.092163 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.096256 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.096288 IP 192.168.0.115.58803 > 142.251.42.66.443: UDP, length 40
12:15:45.097094 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 723
12:15:45.097649 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 692
12:15:45.099437 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:45.110928 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:45.118839 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [.], ack 2805, win 501, options [nop,nop,TS val 1544064719 ecr 187502209], le
ngth 0
12:15:45.120050 IP 192.168.0.115.44636 > 142.251.42.78.443: Flags [P.], seq 517:581, ack 4311, win 501, options [nop,nop,TS val 1243076790 ecr
 1613452755], length 64
12:15:45.120387 IP 192.168.0.115.44636 > 142.251.42.78.443: Flags [P.], seq 581:751, ack 4311, win 501, options [nop,nop,TS val 1243076790 ecr
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:44.944308 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 1357
12:15:44.944332 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 1092
12:15:44.944640 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 785
12:15:44.945712 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 674
12:15:44.946586 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.948213 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.949048 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:44.949113 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.949158 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.949330 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.949532 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.950743 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.950784 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.950817 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.950956 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951004 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951145 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951262 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951465 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 38
12:15:44.951503 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951565 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951684 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951758 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951798 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951853 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.951943 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952056 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952237 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952288 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952409 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952448 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952507 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952626 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952724 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.952802 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.953127 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.953266 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.953344 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:44.832024 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.853179 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.853234 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.860498 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [P.], seq 852:1278, ack 1403, win 501, options [nop,nop,TS val 1544064461 ecr
 187501836], length 426: HTTP: POST /gts1c3 HTTP/1.1
12:15:44.862336 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 110
12:15:44.862395 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 67
12:15:44.863343 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 661
12:15:44.863645 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [S], seq 1147360883, win 64240, options [mss 1460,sackOK,TS val 1544064464 ec
r 0,nop,wscale 7], length 0
12:15:44.866102 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [.], ack 2120284675, win 502, options [nop,nop,TS val 1544064467 ecr 20288875
50], length 0
12:15:44.866309 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [P.], seq 0:426, ack 1, win 502, options [nop,nop,TS val 1544064467 ecr 20288
87550], length 426: HTTP: POST /gts1c3 HTTP/1.1
12:15:44.869276 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 37
12:15:44.870826 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 31
12:15:44.870940 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 32
12:15:44.871030 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 32
12:15:44.872203 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 32
12:15:44.927258 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [.], ack 2104, win 501, options [nop,nop,TS val 1544064528 ecr 187502018], le
ngth 0
12:15:44.927959 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [P.], seq 517:581, ack 4322, win 501, options [nop,nop,TS val 2250039834 ecr
 2531519901], length 64
12:15:44.928123 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [P.], seq 581:751, ack 4322, win 501, options [nop,nop,TS val 2250039835 ecr
 2531519901], length 170
12:15:44.929917 IP 192.168.0.115.38348 > 142.250.192.3.80: Flags [.], ack 702, win 501, options [nop,nop,TS val 1544064530 ecr 2028887614], le
ngth 0
12:15:44.930479 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [.], ack 4936, win 501, options [nop,nop,TS val 2250039837 ecr 2531520053],
length 0
12:15:44.930552 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [.], ack 4967, win 501, options [nop,nop,TS val 2250039837 ecr 2531520053],
length 0
12:15:44.931559 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [P.], seq 751:782, ack 4967, win 501, options [nop,nop,TS val 2250039838 ecr
 2531520053], length 31
12:15:44.931679 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [P.], seq 517:581, ack 4323, win 501, options [nop,nop,TS val 2250039838 ecr
 971904754], length 64
12:15:44.931841 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [P.], seq 581:768, ack 4323, win 501, options [nop,nop,TS val 2250039838 ecr
 971904754], length 187
12:15:44.933614 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [.], ack 4937, win 501, options [nop,nop,TS val 2250039840 ecr 971904907], l
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
length 0
12:15:44.710681 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 3890:3936, ack 1170, win 501, options [nop,nop,TS val 3427210848
ecr 587168828], length 46
12:15:44.710886 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 1245, win 501, options [nop,nop,TS val 3427210848 ecr 587168828],
length 0
12:15:44.710891 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 1553, win 501, options [nop,nop,TS val 3427210848 ecr 587168828],
length 0
12:15:44.710898 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 1649, win 501, options [nop,nop,TS val 3427210848 ecr 587168828],
length 0
12:15:44.713178 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [S], seq 2362265248, win 64240, options [mss 1460,sackOK,TS val 2250039620 e
cr 0,nop,wscale 7], length 0
12:15:44.714354 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [S], seq 4044908289, win 64240, options [mss 1460,sackOK,TS val 2250039621 e
cr 0,nop,wscale 7], length 0
12:15:44.715198 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [.], ack 3273701882, win 502, options [nop,nop,TS val 2250039622 ecr 2531519
838], length 0
12:15:44.716535 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [.], ack 1572572572, win 502, options [nop,nop,TS val 2250039623 ecr 9719046
90], length 0
12:15:44.716909 IP 192.168.0.115.47332 > 142.251.42.35.443: Flags [P.], seq 0:517, ack 1, win 502, options [nop,nop,TS val 2250039623 ecr 2531
519838], length 517
12:15:44.717097 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 1724, win 501, options [nop,nop,TS val 3427210855 ecr 587168834],
length 0
12:15:44.717107 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 1813, win 501, options [nop,nop,TS val 3427210855 ecr 587168834],
length 0
12:15:44.717108 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 2086, win 501, options [nop,nop,TS val 3427210855 ecr 587168834],
length 0
12:15:44.717220 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 2132, win 501, options [nop,nop,TS val 3427210855 ecr 587168834],
length 0
12:15:44.718587 IP 192.168.0.115.47344 > 142.251.42.35.443: Flags [P.], seq 0:517, ack 1, win 502, options [nop,nop,TS val 2250039625 ecr 9719
04690], length 517
12:15:44.718788 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 3936:3982, ack 2132, win 501, options [nop,nop,TS val 3427210856
ecr 587168834], length 46
12:15:44.722344 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.722358 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.744643 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 2601, win 501, options [nop,nop,TS val 3427210882 ecr 587168862],
length 0
12:15:44.744660 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 3982:4028, ack 2601, win 501, options [nop,nop,TS val 3427210882
ecr 587168862], length 46
12:15:44.745599 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [.], ack 1403, win 501, options [nop,nop,TS val 1544064346 ecr 187501836], le
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:44.498794 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.498994 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.499199 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.499315 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.504410 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506149 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506241 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506359 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506566 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506664 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506735 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.506958 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507032 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507126 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507208 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507339 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507440 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507547 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.507767 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.508208 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.508435 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.508603 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.510234 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.510339 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.510432 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.515030 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 654
12:15:44.530507 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 42
12:15:44.552138 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 34
12:15:44.613511 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 1357
12:15:44.616940 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.617050 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 649
12:15:44.619536 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 35
12:15:44.620927 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 33
12:15:44.627312 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.627332 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.647674 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.647695 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 40
12:15:44.677642 IP 192.168.0.115.48593 > 142.251.42.35.443: UDP, length 42
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:15:44.132295 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 67
12:15:44.142519 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 712
12:15:44.143307 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 31
12:15:44.166348 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 32
12:15:44.360630 IP 192.168.0.115.59046 > 142.250.192.132.443: UDP, length 36
12:15:44.388884 IP 192.168.0.115.40533 > 192.168.0.1.53: 65074+ [1au] A? incoming.telemetry.mozilla.org. (59)
12:15:44.389085 IP 192.168.0.115.47584 > 192.168.0.1.53: 45827+ [1au] AAAA? incoming.telemetry.mozilla.org. (59)
12:15:44.391299 IP 192.168.0.115.42314 > 192.168.0.1.53: 58562+ [1au] AAAA? prod.ingestion-edge.prod.dataops.mozgcp.net. (72)
12:15:44.392050 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [S], seq 3360492081, win 64240, options [mss 1460,sackOK,TS val 3427210530
ecr 0,nop,wscale 7], length 0
12:15:44.394717 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 2469793283, win 502, options [nop,nop,TS val 3427210532 ecr 587168
511], length 0
12:15:44.394996 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 0:517, ack 1, win 502, options [nop,nop,TS val 3427210533 ecr 587
168511], length 517
12:15:44.457468 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 157, win 501, options [nop,nop,TS val 3427210595 ecr 587168574], l
ength 0
12:15:44.458133 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 517:568, ack 157, win 501, options [nop,nop,TS val 3427210596 ecr
 587168574], length 51
12:15:44.458757 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 568:745, ack 157, win 501, options [nop,nop,TS val 3427210596 ecr
 587168574], length 177
12:15:44.458820 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 745:1104, ack 157, win 501, options [nop,nop,TS val 3427210596 ec
r 587168574], length 359
12:15:44.458869 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 1104:1528, ack 157, win 501, options [nop,nop,TS val 3427210596 e
cr 587168574], length 424
12:15:44.459121 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 1528:1661, ack 157, win 501, options [nop,nop,TS val 3427210597 e
cr 587168574], length 133
12:15:44.459160 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 1661:2013, ack 157, win 501, options [nop,nop,TS val 3427210597 e
cr 587168574], length 352
12:15:44.459770 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 2013:2141, ack 157, win 501, options [nop,nop,TS val 3427210597 e
cr 587168574], length 128
12:15:44.459800 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 2141:2646, ack 157, win 501, options [nop,nop,TS val 3427210597 e
cr 587168574], length 505
12:15:44.460237 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 226, win 501, options [nop,nop,TS val 3427210598 ecr 587168577], l
ength 0
12:15:44.460310 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [P.], seq 2646:2684, ack 226, win 501, options [nop,nop,TS val 3427210598 e
cr 587168577], length 38
12:15:44.460626 IP 192.168.0.115.58732 > 34.120.208.123.443: Flags [.], ack 264, win 501, options [nop,nop,TS val 3427210598 ecr 587168577], l
ength 0
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -n 192.168.0.115
tcpdump: syntax error in filter expression: syntax error
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -n src 192.168.0.115
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:15:37.049886 ARP, Request who-has 192.168.0.1 tell 192.168.0.115, length 28
12:15:42.547024 IP 192.168.0.115.53025 > 192.168.0.1.53: 60211+ [1au] A? www.google.com. (43)
12:15:42.547213 IP 192.168.0.115.60322 > 192.168.0.1.53: 47916+ [1au] AAAA? www.google.com. (43)
12:15:42.564146 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 1357
12:15:42.567547 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.580961 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.580989 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.606378 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.606402 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.626006 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 42
12:15:42.630699 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 43
12:15:42.631880 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 44
12:15:42.634025 IP 192.168.0.115.35616 > 192.168.0.1.53: 3863+ [1au] A? ocsp.pki.goog. (42)
12:15:42.634111 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.634139 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.634153 IP 192.168.0.115.38673 > 192.168.0.1.53: 25353+ [1au] AAAA? ocsp.pki.goog. (42)
12:15:42.635963 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [S], seq 4171914047, win 64240, options [mss 1460,sackOK,TS val 1544062237 ec
r 0,nop,wscale 7], length 0
12:15:42.638472 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [.], ack 2090364198, win 502, options [nop,nop,TS val 1544062239 ecr 18749972
9], length 0
12:15:42.638619 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [P.], seq 0:426, ack 1, win 502, options [nop,nop,TS val 1544062239 ecr 18749
9729], length 426: HTTP: POST /gts1c3 HTTP/1.1
12:15:42.661388 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.661443 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.686599 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.686658 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.709638 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.709691 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 40
12:15:42.713759 IP 192.168.0.115.38338 > 142.250.192.3.80: Flags [.], ack 702, win 501, options [nop,nop,TS val 1544062314 ecr 187499804], len
gth 0
12:15:42.721921 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 108
12:15:42.721961 IP 192.168.0.115.46740 > 142.250.192.132.443: UDP, length 67
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
ength 0
12:14:51.265370 IP 34.120.208.123.443 > 192.168.0.115.55670: Flags [F.], seq 1, ack 77, win 474, options [nop,nop,TS val 2702799256 ecr 342715
7401], length 0
12:14:51.265371 IP 34.120.208.123.443 > 192.168.0.115.55670: Flags [.], ack 78, win 474, options [nop,nop,TS val 2702799256 ecr 3427157401], l
ength 0
12:14:51.265413 IP 192.168.0.115.55670 > 34.120.208.123.443: Flags [.], ack 2, win 501, options [nop,nop,TS val 3427157403 ecr 2702799256], le
ngth 0
12:14:51.435552 ARP, Request who-has 192.168.0.237 tell 192.168.0.1, length 46
12:14:51.566939 ARP, Request who-has 192.168.0.181 tell 192.168.0.114, length 46
12:14:51.677208 IP 192.168.0.173.5353 > 224.0.0.251.5353: 55028 PTR (QM)? _arduino._tcp.local. (37)
12:14:51.677490 IP6 fe80::4c0c:70b:9722:50e2.5353 > ff02::fb.5353: 55028 PTR (QM)? _arduino._tcp.local. (37)
12:14:51.677724 IP 192.168.0.241.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
12:14:51.677732 IP 192.168.0.148.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
12:14:51.678026 IP6 fe80::d08:56ec:5b45:e946.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
12:14:51.678039 IP6 fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
12:14:51.888825 IP 192.168.0.147.54402 > 239.255.255.250.1900: UDP, length 175
12:14:52.193832 IP6 fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
12:14:52.194167 IP6 fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
12:14:52.194633 IP6 fe80::d08:56ec:5b45:e946.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
12:14:52.478575 ARP, Request who-has 192.168.0.181 tell 192.168.0.114, length 46
12:14:52.889874 IP 192.168.0.147.54402 > 239.255.255.250.1900: UDP, length 175
12:14:53.488036 ARP, Request who-has 192.168.0.181 tell 192.168.0.114, length 46
^C
24 packets captured
24 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# ifconfig
enp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.115  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::192e:4c94:cfd4:5894  prefixlen 64  scopeid 0x20<link>
        ether 04:0e:3c:1a:5c:ad  txqueuelen 1000  (Ethernet)
        RX packets 45820  bytes 8408261 (8.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5746  bytes 632580 (632.5 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
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
12:07:06.414888 IP _gateway.56935 > 239.255.255.250.1900: UDP, length 448
^C
40 packets captured
42 packets received by filter
2 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump udp and src port 1900
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
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump udp and src port 63111
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
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# 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
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump udp and dst port 1900
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:09:12.612120 IP 192.168.0.184.56692 > 239.255.255.250.1900: UDP, length 175
12:09:12.931422 IP 192.168.0.171.50787 > 239.255.255.250.1900: UDP, length 175
12:09:13.531237 IP 192.168.0.114.63910 > 239.255.255.250.1900: UDP, length 175
12:09:13.616188 IP 192.168.0.184.56692 > 239.255.255.250.1900: UDP, length 175
12:09:13.940277 IP 192.168.0.171.50787 > 239.255.255.250.1900: UDP, length 175
12:09:14.955699 IP 192.168.0.171.50787 > 239.255.255.250.1900: UDP, length 175
12:09:15.971092 IP 192.168.0.171.50787 > 239.255.255.250.1900: UDP, length 175
12:09:24.995085 IP 192.168.0.125.57199 > 239.255.255.250.1900: UDP, length 176
12:09:25.960224 IP 192.168.0.171.50788 > 239.255.255.250.1900: UDP, length 175
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:06:41.579320 IP 192.168.0.194.64261 > 239.255.255.250.1900: UDP, length 175
12:06:41.833679 IP 192.168.0.110.55316 > 239.255.255.250.1900: UDP, length 175
^C
22 packets captured
22 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump port 1900
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:06:55.282611 IP 192.168.0.187.52797 > 239.255.255.250.1900: UDP, length 175
12:06:55.547169 IP 192.168.0.42.56375 > 239.255.255.250.1900: UDP, length 176
12:06:56.297965 IP 192.168.0.187.52797 > 239.255.255.250.1900: UDP, length 175
12:06:56.551647 IP 192.168.0.42.56375 > 239.255.255.250.1900: UDP, length 176
12:06:56.943337 IP 192.168.0.187.52798 > 239.255.255.250.1900: UDP, length 167
12:06:57.313339 IP 192.168.0.187.52797 > 239.255.255.250.1900: UDP, length 175
12:06:57.563412 IP 192.168.0.42.56375 > 239.255.255.250.1900: UDP, length 176
12:06:57.953434 IP 192.168.0.187.52798 > 239.255.255.250.1900: UDP, length 167
12:06:58.574734 IP 192.168.0.42.56375 > 239.255.255.250.1900: UDP, length 176
12:06:58.969103 IP 192.168.0.187.52798 > 239.255.255.250.1900: UDP, length 167
12:06:59.984499 IP 192.168.0.187.52798 > 239.255.255.250.1900: UDP, length 167
12:07:00.312334 IP 192.168.0.172.54515 > 239.255.255.250.1900: UDP, length 176
12:07:00.529838 IP 192.168.0.111.60909 > 239.255.255.250.1900: UDP, length 176
12:07:01.323663 IP 192.168.0.172.54515 > 239.255.255.250.1900: UDP, length 176
12:07:01.428569 IP 192.168.0.239.54615 > 239.255.255.250.1900: UDP, length 176
12:07:01.539519 IP 192.168.0.111.60909 > 239.255.255.250.1900: UDP, length 176
12:07:02.325628 IP 192.168.0.172.54515 > 239.255.255.250.1900: UDP, length 176
12:07:02.436420 IP 192.168.0.239.54615 > 239.255.255.250.1900: UDP, length 176
12:07:02.542964 IP 192.168.0.111.60909 > 239.255.255.250.1900: UDP, length 176
12:07:03.255049 IP 192.168.0.114.58960 > 239.255.255.250.1900: UDP, length 176
12:07:03.297587 IP 192.168.0.247.49206 > 239.255.255.250.1900: UDP, length 176
12:07:03.333116 IP 192.168.0.172.54515 > 239.255.255.250.1900: UDP, length 176
12:07:03.452199 IP 192.168.0.239.54615 > 239.255.255.250.1900: UDP, length 176
12:07:03.545714 IP 192.168.0.111.60909 > 239.255.255.250.1900: UDP, length 176
12:07:04.256554 IP 192.168.0.114.58960 > 239.255.255.250.1900: UDP, length 176
12:07:04.310011 IP 192.168.0.247.49206 > 239.255.255.250.1900: UDP, length 176
12:07:04.467224 IP 192.168.0.239.54615 > 239.255.255.250.1900: UDP, length 176
12:07:04.579109 IP 192.168.0.235.63111 > 239.255.255.250.1900: UDP, length 175
12:07:05.270383 IP 192.168.0.114.58960 > 239.255.255.250.1900: UDP, length 176
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
12:04:21.929912 ARP, Reply 192.168.0.115 is-at 04:0e:3c:1a:5c:ad, length 28
12:04:21.930005 ARP, Reply 192.168.0.115 is-at 04:0e:3c:1a:5c:ad, length 28
^C
5 packets captured
5 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -n dst 192.168.0.115
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:05:04.090274 ARP, Request who-has 192.168.0.115 tell 192.168.0.1, length 46
12:05:04.090365 ARP, Request who-has 192.168.0.115 tell 192.168.0.1, length 46
12:05:21.874791 ARP, Request who-has 192.168.0.115 tell 192.168.0.1, length 46
12:05:21.875387 ARP, Request who-has 192.168.0.115 tell 192.168.0.1, length 46
12:05:34.778085 ARP, Request who-has 192.168.0.115 tell 192.168.0.1, length 46
12:05:35.269223 IP 192.168.0.1.34826 > 192.168.0.115.137: UDP, length 50
12:05:35.280883 IP 192.168.0.1.56491 > 192.168.0.115.137: UDP, length 50
12:05:40.314696 ARP, Reply 192.168.0.1 is-at ac:15:a2:b9:9e:29, length 46
^C
8 packets captured
8 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -n port 1900
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:06:37.477268 IP 192.168.0.154.60837 > 239.255.255.250.1900: UDP, length 176
12:06:37.971101 IP 192.168.0.112.64148 > 239.255.255.250.1900: UDP, length 175
12:06:38.007156 IP 192.168.0.248.63881 > 239.255.255.250.1900: UDP, length 175
12:06:38.112639 IP 192.168.0.188.61755 > 239.255.255.250.1900: UDP, length 175
12:06:38.164787 IP 192.168.0.110.55315 > 239.255.255.250.1900: UDP, length 175
12:06:38.977092 IP 192.168.0.112.64148 > 239.255.255.250.1900: UDP, length 175
12:06:39.011964 IP 192.168.0.248.63881 > 239.255.255.250.1900: UDP, length 175
12:06:39.128209 IP 192.168.0.188.61755 > 239.255.255.250.1900: UDP, length 175
12:06:39.166932 IP 192.168.0.110.55315 > 239.255.255.250.1900: UDP, length 175
12:06:39.542216 IP 192.168.0.141.53102 > 239.255.255.250.1900: UDP, length 175
12:06:39.577567 IP 192.168.0.194.64261 > 239.255.255.250.1900: UDP, length 175
12:06:39.982131 IP 192.168.0.112.64148 > 239.255.255.250.1900: UDP, length 175
12:06:40.026959 IP 192.168.0.248.63881 > 239.255.255.250.1900: UDP, length 175
12:06:40.143888 IP 192.168.0.188.61755 > 239.255.255.250.1900: UDP, length 175
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 921  bytes 102974 (102.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 921  bytes 102974 (102.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -e -v -n
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:03:12.836587 f4:39:09:49:07:05 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 57593, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.184.58818 > 239.255.255.250.1900: UDP, length 175
12:03:12.914646 a0:8c:fd:c4:b6:d4 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 44236, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.171.60344 > 239.255.255.250.1900: UDP, length 175
12:03:13.524933 04:0e:3c:19:2e:0f > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 29890, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.114.52294 > 239.255.255.250.1900: UDP, length 175
12:03:13.605553 d4:be:d9:cc:01:69 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
168.0.183 tell 192.168.0.129, length 46
12:03:13.777995 04:0e:3c:19:28:92 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
168.0.150 tell 192.168.0.138, length 46
12:03:13.837365 f4:39:09:49:07:05 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 57594, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.184.58818 > 239.255.255.250.1900: UDP, length 175
12:03:13.923114 a0:8c:fd:c4:b6:d4 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 44237, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.171.60344 > 239.255.255.250.1900: UDP, length 175
12:03:14.599702 d4:be:d9:cc:01:69 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
168.0.183 tell 192.168.0.129, length 46
12:03:14.623674 04:0e:3c:19:28:92 > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
168.0.150 tell 192.168.0.138, length 46
12:03:14.692271 04:0e:3c:19:2b:2e > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
168.0.224 tell 192.168.0.141, length 46
12:03:14.938600 a0:8c:fd:c4:b6:d4 > 01:00:5e:7f:ff:fa, ethertype IPv4 (0x0800), length 217: (tos 0x0, ttl 1, id 44238, offset 0, flags [none],
 proto UDP (17), length 203)
    192.168.0.171.60344 > 239.255.255.250.1900: UDP, length 175
12:03:15.067050 f4:39:09:49:07:cd > ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), length 60: Ethernet (len 6), IPv4 (len 4), Request who-has 192.
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
0/0 (12)
11:56:53.428804 IP6 (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)
11:56:53.563975 IP (tos 0x0, ttl 64, id 37497, offset 0, flags [DF], proto UDP (17), length 354)
    192.168.0.1.55083 > 192.168.0.255.20002: UDP, length 326
11:56:53.627911 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 86) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
p sum ok] 0*- [0q] 1/0/0 lab1006-HP-280-G4-MT-Business-PC.local. (Cache flush) AAAA fe80::92cc:9f56:6283:b766 (78)
11:56:53.656577 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.224 tell 192.168.0.234, length 46
11:56:53.674275 IP (tos 0x0, ttl 128, id 2958, offset 0, flags [none], proto UDP (17), length 78)
    192.168.0.175.137 > 192.168.0.255.137: UDP, length 50
11:56:53.772608 IP6 (flowlabel 0xabf25, hlim 255, next-header UDP (17) payload length: 172) fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: [udp
 sum ok] 0*- [0q] 2/0/0 6.f.4.0.4.3.2.0.b.0.d.8.2.7.7.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa. (Cache flush) PTR lab1006-HP-280-G4-MT-Busin
ess-PC.local., lab1006-HP-280-G4-MT-Business-PC.local. (Cache flush) AAAA fe80::c772:8d0b:234:4f6 (164)
11:56:53.778389 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.179 tell 192.168.0.1, length 46
11:56:53.858580 IP (tos 0x0, ttl 1, id 56440, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.147.61590 > 239.255.255.250.1900: UDP, length 175
11:56:54.045269 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 92) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
p sum ok] 0 [1n] ANY (QM)? lab1006-HP-280-G4-MT-Business-PC.local. (84)
11:56:54.045291 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:54.045832 IP6 (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)
11:56:54.081638 IP (tos 0x0, ttl 1, id 60169, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.130.51801 > 239.255.255.250.1900: UDP, length 175
11:56:54.135188 IP (tos 0x0, ttl 1, id 25401, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.155.56042 > 239.255.255.250.1900: UDP, length 175
11:56:54.235545 IP (tos 0x0, ttl 1, id 60750, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.187.60811 > 239.255.255.250.1900: UDP, length 175
11:56:54.295842 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 92) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
p sum ok] 0 [1n] ANY (QM)? lab1006-HP-280-G4-MT-Business-PC.local. (84)
11:56:54.295864 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:54.296646 IP6 (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)
11:56:54.434992 IP (tos 0x0, ttl 128, id 2959, offset 0, flags [none], proto UDP (17), length 78)
    192.168.0.175.137 > 192.168.0.255.137: UDP, length 50
11:56:54.507640 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.166 tell 192.168.0.212, length 46
11:56:54.545905 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 92) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
p6 sum ok] ICMP6, multicast listener report v2, 1 group record(s) [gaddr ff02::fb to_ex, 0 source(s)]
11:56:52.751349 IP (tos 0x10, ttl 128, id 0, offset 0, flags [none], proto UDP (17), length 328)
    0.0.0.0.68 > 255.255.255.255.67: BOOTP/DHCP, Request from 04:0e:3c:1a:5c:74, length 300, xid 0x99988c70, secs 3, Flags [none]
          Client-Ethernet-Address 04:0e:3c:1a:5c:74
          Vendor-rfc1048 Extensions
            Magic Cookie 0x63825363
            DHCP-Message Option 53, length 1: Discover
            Hostname Option 12, length 32: "lab1006-HP-280-G4-MT-Business-PC"
            Parameter-Request Option 55, length 16:
              Subnet-Mask, BR, Time-Zone, Default-Gateway
              Domain-Name, Domain-Name-Server, Option 119, Hostname
              Netbios-Name-Server, Netbios-Scope, MTU, Classless-Static-Route
              NTP, Classless-Static-Route-Microsoft, Static-Route, Option 252
11:56:52.779349 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.183 tell 192.168.0.231, length 46
11:56:52.853687 IP (tos 0x0, ttl 1, id 56439, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.147.61590 > 239.255.255.250.1900: UDP, length 175
11:56:52.885119 IP6 (flowlabel 0xabf25, hlim 255, next-header UDP (17) payload length: 53) fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: [udp
sum ok] 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
11:56:52.885252 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:52.885917 IP6 (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)
11:56:52.908993 IP (tos 0x0, ttl 1, id 24133, offset 0, flags [none], proto UDP (17), length 65)
    192.168.0.173.5353 > 224.0.0.251.5353: 881 PTR (QM)? _arduino._tcp.local. (37)
11:56:52.909015 IP (tos 0x0, ttl 1, id 17739, offset 0, flags [none], proto UDP (17), length 40)
    192.168.0.241.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
11:56:52.909648 IP6 (flowlabel 0x925be, hlim 1, next-header UDP (17) payload length: 45) fe80::4c0c:70b:9722:50e2.5353 > ff02::fb.5353: [udp s
um ok] 881 PTR (QM)? _arduino._tcp.local. (37)
11:56:52.909661 IP (tos 0x0, ttl 1, id 13814, 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)
11:56:52.909759 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:52.910129 IP6 (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)
11:56:52.927143 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 92) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
p sum ok] 0 [1n] ANY (QM)? lab1006-HP-280-G4-MT-Business-PC.local. (84)
11:56:52.927254 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
33 packets received by filter
0 packets dropped by kernel
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -v -n
tcpdump: listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:56:51.778995 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.183 tell 192.168.0.231, length 46
11:56:51.783048 IP6 (hlim 1, next-header Options (0) payload length: 36) fe80::c772:8d0b:234:4f6 > ff02::16: HBH (rtalert: 0x0000) (padn) [icm
p6 sum ok] ICMP6, multicast listener report v2, 1 group record(s) [gaddr ff02::1:ff34:4f6 to_ex, 0 source(s)]
11:56:51.794888 IP6 (hlim 1, next-header Options (0) payload length: 36) fe80::c772:8d0b:234:4f6 > ff02::16: HBH (rtalert: 0x0000) (padn) [icm
p6 sum ok] ICMP6, multicast listener report v2, 1 group record(s) [gaddr ff02::fb to_ex, 0 source(s)]
11:56:51.819994 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.166 tell 192.168.0.212, length 46
11:56:51.848452 IP (tos 0x0, ttl 1, id 56438, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.147.61590 > 239.255.255.250.1900: UDP, length 175
11:56:51.884501 IP6 (flowlabel 0xabf25, hlim 255, next-header UDP (17) payload length: 53) fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: [udp
sum ok] 0 [2q] PTR (QM)? _ipps._tcp.local. PTR (QM)? _ipp._tcp.local. (45)
11:56:51.884598 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:51.885193 IP6 (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)
11:56:51.891718 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.0.224 tell 192.168.0.173, length 46
11:56:51.950039 IP (tos 0x0, ttl 1, id 12232, offset 0, flags [none], proto UDP (17), length 204)
    192.168.0.103.54594 > 239.255.255.250.1900: UDP, length 176
11:56:51.951603 IP6 (flowlabel 0xabf25, hlim 255, next-header UDP (17) payload length: 184) fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: [udp
 sum ok] 0 [2q] [2n] ANY (QM)? 6.f.4.0.4.3.2.0.b.0.d.8.2.7.7.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa. ANY (QM)? lab1006-HP-280-G4-MT-Busine
ss-PC.local. (176)
11:56:51.951624 IP6 (hlim 1, next-header UDP (17) payload length: 20) fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: [udp sum ok] 0*- [0q] 0/
0/0 (12)
11:56:51.952306 IP6 (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)
11:56:51.952327 IP6 (flowlabel 0xb3690, hlim 255, next-header UDP (17) payload length: 86) fe80::92cc:9f56:6283:b766.5353 > ff02::fb.5353: [ud
p sum ok] 0*- [0q] 1/0/0 lab1006-HP-280-G4-MT-Business-PC.local. (Cache flush) AAAA fe80::92cc:9f56:6283:b766 (78)
11:56:51.991454 IP6 (flowlabel 0x11162, hlim 255, next-header ICMPv6 (58) payload length: 8) fe80::c772:8d0b:234:4f6 > ff02::2: [icmp6 sum ok]
 ICMP6, router solicitation, length 8
11:56:52.068699 IP (tos 0x0, ttl 1, id 60167, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.130.51801 > 239.255.255.250.1900: UDP, length 175
11:56:52.132250 IP (tos 0x0, ttl 1, id 25399, offset 0, flags [none], proto UDP (17), length 203)
    192.168.0.155.56042 > 239.255.255.250.1900: UDP, length 175
11:56:52.202772 IP6 (flowlabel 0xabf25, hlim 255, next-header UDP (17) payload length: 184) fe80::c772:8d0b:234:4f6.5353 > ff02::fb.5353: [udp
 sum ok] 0 [2q] [2n] ANY (QM)? 6.f.4.0.4.3.2.0.b.0.d.8.2.7.7.c.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa. ANY (QM)? lab1006-HP-280-G4-MT-Busine
```

```
val 910409383 ecr 4071336851,nop,wscale 7], length 0
11:56:15.797989 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 4071337094 ecr 910409383], lengt
h 0
11:56:15.798187 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 4071337094 ecr 910409
383], length 87: HTTP: GET / HTTP/1.1
11:56:16.038976 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [.], ack 88, win 506, options [nop,nop,TS val 910409628 ecr 4071337094], leng
th 0
11:56:16.038999 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [P.], seq 1:149, ack 88, win 506, options [nop,nop,TS val 910409628 ecr 40713
37094], length 148: HTTP: HTTP/1.1 204 No Content
11:56:16.039040 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [.], ack 149, win 501, options [nop,nop,TS val 4071337335 ecr 910409628], len
gth 0
11:56:16.039262 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [F.], seq 88, ack 149, win 501, options [nop,nop,TS val 4071337335 ecr 910409
628], length 0
11:56:16.039607 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 910409628 ecr 4071337
094], length 0
11:56:16.039636 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [.], ack 150, win 501, options [nop,nop,TS val 4071337335 ecr 910409628], len
gth 0
11:56:16.247663 IP 192.168.0.243.59368 > 239.255.255.250.1900: UDP, length 175
11:56:16.280422 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [.], ack 89, win 506, options [nop,nop,TS val 910409869 ecr 4071337335], leng
th 0
11:56:16.332048 IP 192.168.0.211.62996 > 239.255.255.250.1900: UDP, length 175
11:56:16.340927 IP 192.168.0.211.62995 > 239.255.255.250.1900: UDP, length 175
11:56:16.640479 IP 192.168.0.173.5353 > 224.0.0.251.5353: 29573 PTR (QM)? _arduino._tcp.local. (37)
11:56:16.640502 IP6 fe80::4c0c:70b:9722:50e2.5353 > ff02::fb.5353: 29573 PTR (QM)? _arduino._tcp.local. (37)
11:56:16.640526 IP 192.168.0.148.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
11:56:16.640541 IP 192.168.0.241.5353 > 224.0.0.251.5353: 0*- [0q] 0/0/0 (12)
11:56:16.640618 IP6 fe80::d08:56ec:5b45:e946.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
11:56:16.640772 IP6 fe80::98b4:47fb:4996:5056.5353 > ff02::fb.5353: 0*- [0q] 0/0/0 (12)
11:56:17.253111 IP 192.168.0.243.59368 > 239.255.255.250.1900: UDP, length 175
11:56:17.347235 IP 192.168.0.211.62996 > 239.255.255.250.1900: UDP, length 175
11:56:17.347258 IP 192.168.0.211.62995 > 239.255.255.250.1900: UDP, length 175
11:56:17.411655 IP 192.168.0.203.62289 > 239.255.255.250.1900: UDP, length 175
11:56:18.362522 IP 192.168.0.211.62995 > 239.255.255.250.1900: UDP, length 175
11:56:18.362545 IP 192.168.0.211.62996 > 239.255.255.250.1900: UDP, length 175
11:56:18.421069 IP 192.168.0.203.62289 > 239.255.255.250.1900: UDP, length 175
11:56:19.373665 IP 192.168.0.211.62996 > 239.255.255.250.1900: UDP, length 175
11:56:19.435671 IP 192.168.0.203.62289 > 239.255.255.250.1900: UDP, length 175
11:56:19.559396 IP 192.168.0.165.65504 > 239.255.255.250.1900: UDP, length 176
```

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ tcpdump -D
1.enp3s0 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.nflog (Linux netfilter log (NFLOG) interface)
5.nfqueue (Linux netfilter queue (NFQUEUE) interface)
6.usbmon1 (USB bus number 1)
7.usbmon2 (USB bus number 2)
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ tcpdump -n
tcpdump: enp3s0: You don't have permission to capture on that device
(socket: Operation not permitted)
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ tcpdump -v -n
tcpdump: enp3s0: You don't have permission to capture on that device
(socket: Operation not permitted)
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ sudo su
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -D
1.enp3s0 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.nflog (Linux netfilter log (NFLOG) interface)
5.nfqueue (Linux netfilter queue (NFQUEUE) interface)
6.usbmon1 (USB bus number 1)
7.usbmon2 (USB bus number 2)
root@lab1006-HP-280-G4-MT-Business-PC:/home/lab1006/Desktop# tcpdump -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:56:15.232281 IP 192.168.0.243.59368 > 239.255.255.250.1900: UDP, length 175
11:56:15.282960 ARP, Request who-has 192.168.0.183 tell 192.168.0.231, length 46
11:56:15.336266 IP 192.168.0.211.62995 > 239.255.255.250.1900: UDP, length 175
11:56:15.555942 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [S], seq 4265862487, win 64240, options [mss 1460,sackOK,TS val 4071336851 ec
r 0,nop,wscale 7], length 0
11:56:15.797918 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [S.], seq 3896500190, ack 4265862488, win 64768, options [mss 1420,sackOK,TS
val 910409383 ecr 4071336851,nop,wscale 7], length 0
11:56:15.797989 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 4071337094 ecr 910409383], lengt
h 0
11:56:15.798187 IP 192.168.0.115.38528 > 35.224.170.84.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 4071337094 ecr 910409
383], length 87: HTTP: GET / HTTP/1.1
11:56:16.038976 IP 35.224.170.84.80 > 192.168.0.115.38528: Flags [.], ack 88, win 506, options [nop,nop,TS val 910409628 ecr 4071337094], leng
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/Desktop/hashfile/newfile12.txt
mv: missing destination file operand after '/home/Desktop/hashfile/newfile12.txt'
Try 'mv --help' for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/Desktop/hashfile/newfile12.txt /innerfile
mv: cannot stat '/home/Desktop/hashfile/newfile12.txt': No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/Desktop/hashfile/newfile12.txt /innerfolder
mv: cannot stat '/home/Desktop/hashfile/newfile12.txt': No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/Desktop/result2.txt /hashfile
mv: cannot stat '/home/Desktop/result2.txt': No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/lab1006/Desktop/result2.txt /hashfile
mv: cannot move '/home/lab1006/Desktop/result2.txt' to '/hashfile': Permission denied
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/lab1006/Desktop/hashfile/newfile12.txt /innerfolder
mv: cannot move '/home/lab1006/Desktop/hashfile/newfile12.txt' to '/innerfolder': Permission denied
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ sudo apt-get install tcpdump
[sudo] password for lab1006:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-125 linux-hwe-5.4-headers-5.4.0-126 linux-hwe-5.4-headers-5.4.0-128 linux-hwe-5.4-headers-5.4.0-131
  linux-hwe-5.4-headers-5.4.0-135 linux-hwe-5.4-headers-5.4.0-136 linux-hwe-5.4-headers-5.4.0-137 linux-hwe-5.4-headers-5.4.0-144
  linux-hwe-5.4-headers-5.4.0-146 linux-hwe-5.4-headers-5.4.0-147 linux-hwe-5.4-headers-5.4.0-148
Use 'sudo apt autoremove' to remove them.
The following packages will be upgraded:
  tcpdump
1 upgraded, 0 newly installed, 0 to remove and 252 not upgraded.
Need to get 364 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu bionic-updates/main amd64 tcpdump amd64 4.9.3-0ubuntu0.18.04.3 [364 kB]
Fetched 364 kB in 2s (179 kB/s)
(Reading database ... 374132 files and directories currently installed.)
Preparing to unpack .../tcpdump_4.9.3-0ubuntu0.18.04.3_amd64.deb ...
Unpacking tcpdump (4.9.3-0ubuntu0.18.04.3) over (4.9.3-0ubuntu0.18.04.2) ...
Setting up tcpdump (4.9.3-0ubuntu0.18.04.3) ...
Installing new version of config file /etc/apparmor.d/usr.sbin.tcpdump ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ tcpdump -D
1.enp3s0 [Up, Running]
```

```
Activities  Terminal  Wed 12:18
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
/home/lab1006/Desktop/hdfile1.txt
/home/lab1006/Desktop/quatilePlot
/home/lab1006/Desktop/result.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s -x -k result1.txt *
/home/lab1006/Desktop/apiori.py
/home/lab1006/Desktop/asso.py
/home/lab1006/Desktop/bay.py
/home/lab1006/Desktop/bayes.py
/home/lab1006/Desktop/boxplot.py
/home/lab1006/Desktop/apiori.csv
/home/lab1006/Desktop/data.csv
/home/lab1006/Desktop/dataset.csv
/home/lab1006/Desktop/histo.py
/home/lab1006/Desktop/hashlab
/home/lab1006/Desktop/Lab Assignment 5 Hashing.pdf
/home/lab1006/Desktop/quatilePlot.py
/home/lab1006/Desktop/result1.txt
/home/lab1006/Desktop/result2.txt
/home/lab1006/Desktop/Lab Assignment 11 Snort IDS.pdf
/home/lab1006/Desktop/scatter.py
/home/lab1006/Desktop/Lab Assignment 7 TCPDUMP.pdf
/home/lab1006/Desktop/s.python
/home/lab1006/Desktop/Lab Assignment 11 Firewalls using IPTABLES.pdf
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5,sha1,sha256 -r /home/lab1006/Desktop/hashfile>hashsets0.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -a -r -k hashsets0.txt /home/lab1006/Desktop/hashfile
hashdeep: Audit passed
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ touch /home/lab1006/Desktop/hashfile/newfile12.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -a -r -k hashsets0.txt /home/lab1006/Desktop/hashfile
hashdeep: Audit failed
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -v -a -r -k hashsets0.txt /home/lab1006/Desktop/hashfile
hashdeep: Audit failed
           Files matched: 2
Files partially matched: 0
            Files moved: 0
        New files found: 1
  Known files not found: 0
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ mv /home/Desktop/hashfile/newfile12.txt
mv: missing destination file operand after '/home/Desktop/hashfile/newfile12.txt'
```

Activities Terminal Wed 12:18

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
## $ hashdeep -s -k result1.txt 139 71 apiori.csv apiori.py asso.py bayes.py bay.py boxplot.py data.csv dataset.csv DT foldwq hashfile hashlab
 hdfile1.txt histo.py Lab Assignment 11 Firewalls using IPTABLES.pdf Lab Assignment 11 Snort IDS.pdf Lab Assignment 5 Hashing.pdf Lab Assignme
nt 7 TCPDUMP.pdf labhash Old Firefox Data quatilePlot quatilePlot.py result1.txt result2.txt result.txt scatter.py s.python t12_29
##
926,5b4a24394cc807e45eecce4d3c78b7b0,c8b6256d75bbb559a8061f06d14ee52fb8239bd32fcb1457ff2c806e8a28ed96,/home/lab1006/Desktop/apiori.py
153,892edb1f2bc1179f7c57e899bbb238ed,62f28ede8a1650efbebd2d69e1c190804535573b3c46843b6379d259b50d42e7,/home/lab1006/Desktop/boxplot.py
929,2bc246835e080a3892c26d1ac505079f,c778d1ece90974aa0e57616cb7d64bcebc20ba977ba6769ae06acc0a12d4602d,/home/lab1006/Desktop/bayes.py
1835,c726b9a8bff38514d6ee2b06e896b914,a370641fadb06176e015ccc70f16b6a078411b179dbeafe51a760c39a705499a,/home/lab1006/Desktop/asso.py
135,5bc989f19ceefe3af50a1437d2a051ab,53935c6aa290db936914f7d7b6b450150b0e5b2ad091d14037431103c2d79241,/home/lab1006/Desktop/data.csv
92,bac48f7a56f11b362aca191188121d7c,0e30d6e0b5eea5cd625b1a40a4776dfd26378286318c8df45c73765e6c433137,/home/lab1006/Desktop/apiori.csv
352,ebb263293b41391c84bdeb532b178169,8b4c50146171974bc10f03e017e51788586e98aaad5c810cd79038b11cf144e6,/home/lab1006/Desktop/bay.py
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,/home/lab1006/Desktop/hdfile1.txt
147,4147b3c2e6e13b04e251980d186b5959,77c207e79e4f73930d57e44a45d7c8b1272e84f90eb5286b232166d8ad5fc800,/home/lab1006/Desktop/histo.py
368,0d048cab224eb01afae7fb6b7659bb54,428d674827c8b3aefec97d8d0561938deaaddf38d638fb1409dc22bb5ba143e9,/home/lab1006/Desktop/dataset.csv
9,4553dceb0439f8daa82f5e025e8e57d6,b66a511aeb53cea14af7ad2300ed0aa020b74e73fe664f1d3ff3cfe1e2b5f925,/home/lab1006/Desktop/hashlab
0,d41d8cd98f00b204e9800998ecf8427e,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,/home/lab1006/Desktop/quatilePlot
196,f54e281efa4800c06e35e2ab2c23330e,26fac8af6ff03a9dd19dcdd786267a18f396106fa280ff4c7653e133a4015289,/home/lab1006/Desktop/quatilePlot.py
66553,0f16ee5e3d853a3a436ef5429f0147e5,2a5d9f3cad8ff43d4407d1bd7ed185bc2ea56de90f27c60540954d217ad621fb,/home/lab1006/Desktop/Lab Assignment 5
 Hashing.pdf
544,9f1e128198878fc11f0b5ae996c33538,8d9ce361653ec8e70daf1e518af37ac4f96675753bb27fbbc1f28b0a65a02c7f,/home/lab1006/Desktop/result1.txt
384,ab72a33ce65e9bef6be12decf9e0a4d2,191c62c4b122ef21d78cb1c5d2589fca34faa931fbeb3007e1213d769c5ff464,/home/lab1006/Desktop/result.txt
178,720d9dd1dbd89558f038ce4269251f6a,ab12c4573c36595924f2c7efaed2faacb6f065c809874c0c290b7118818a810f,/home/lab1006/Desktop/scatter.py
546,7b3ffb801f90d6113bd88e525d539105,fbb5c7eb925b7d9b952df9c617386a73d08bbcd142efcb57791d832c054f00bb,/home/lab1006/Desktop/result2.txt
229,b5c7a291a75c8878ea72184aabe52993,1eeea14ecb409c193ceadd264ecd210d827fdd31a64eb83c884275df5b7d9a9d,/home/lab1006/Desktop/s.python
93580,2182f6ef78acc795b0fcdfab4cbcee24,a19b910ea44713ba7493458fef0a80e4a43cb23ef20e5c4ff921655b7cfafeb7,/home/lab1006/Desktop/Lab Assignment 1
1 Snort IDS.pdf
92267,f40074c6734a5060aae4882c0e270fec,a7ef59019414702002f59f716e220e99605c4e10a2c6f2505babc1b78dccd9ef,/home/lab1006/Desktop/Lab Assignment 7
 TCPDUMP.pdf
2283012,9085cce548a93c1afed3cbcd742b067b,3350479cfd5cd481e3f2a6579cbcb432f41028af633e437c699a22067c8acc8b,/home/lab1006/Desktop/Lab Assignment
 11 Firewalls using IPTABLES.pdf
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s -m result1.txt *
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -m -k -s result1.txt *
hashdeep: -s: No such file or directory
hashdeep: Unable to load any matching files.
Try `hashdeep -h` for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s -m -k result1.txt *
/home/lab1006/Desktop/hdfile1.txt
/home/lab1006/Desktop/quatilePlot
```

Activities Terminal Wed 12:17

root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop

File Edit View Search Terminal Help

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep *.txt>result1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep *.txt>result2.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ cat result1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep hdfile1.txt result1.txt result.txt
##
0,d41d8cd98f00b204e9800998ecf8427e,e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855,/home/lab1006/Desktop/result1.txt
384,ab72a33ce65e9bef6be12decf9e0a4d2,191c62c4b122ef21d78cb1c5d2589fca34faa931fbeb3007e1213d769c5ff464,/home/lab1006/Desktop/result.txt
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,/home/lab1006/Desktop/hdfile1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ cat result2.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep hdfile1.txt result1.txt result.txt
##
544,9f1e128198878fc11f0b5ae996c33538,8d9ce361653ec8e70daf1e518af37ac4f96675753bb27fbbc1f28b0a65a02c7f,/home/lab1006/Desktop/result1.txt
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,/home/lab1006/Desktop/hdfile1.txt
384,ab72a33ce65e9bef6be12decf9e0a4d2,191c62c4b122ef21d78cb1c5d2589fca34faa931fbeb3007e1213d769c5ff464,/home/lab1006/Desktop/result.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ md5deep -m result1.txt *
md5deep: result1.txt: Unable to find any hashes in file, skipped.
md5deep: Unable to load any matching files.
Try `md5deep -h` for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -m -k result1.txt *
/home/lab1006/Desktop/139: Is a directory
/home/lab1006/Desktop/71: Is a directory
/home/lab1006/Desktop/DT: Is a directory
/home/lab1006/Desktop/foldwq: Is a directory
/home/lab1006/Desktop/hashfile: Is a directory
/home/lab1006/Desktop/hdfile1.txt
/home/lab1006/Desktop/labhash: Is a directory
/home/lab1006/Desktop/Old Firefox Data: Is a directory
/home/lab1006/Desktop/quatilePlot
/home/lab1006/Desktop/result.txt
/home/lab1006/Desktop/t12_29: Is a directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s -k result1.txt *
%%%% HASHDEEP-1.0
```

```
Activities  Terminal  Wed 12:17
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
##
32,3592e22887f8f7fc071f7a981b28534e,30c13a60bdd015bd44da4264f17d92917b0a8736,/home/lab1006/Desktop/hdfile1.txt
384,ab72a33ce65e9bef6be12decf9e0a4d2,93e9d4f123cf32def54b443add1bc1af2f0c1a79,/home/lab1006/Desktop/result.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5 -p 100 hdfile1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5 -p 100 hdfile1.txt
##
32,3592e22887f8f7fc071f7a981b28534e,/home/lab1006/Desktop/hdfile1.txt offset 0-31
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5 -p 16 hdfile1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5 -p 16 hdfile1.txt
##
16,f36814f024f38531da10de19892e1010,/home/lab1006/Desktop/hdfile1.txt offset 0-15
16,02aa3f8004024e036ccd1be066aa594b,/home/lab1006/Desktop/hdfile1.txt offset 16-31
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5 -p 8 hdfile1.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5 -p 8 hdfile1.txt
##
8,5e9a2fa284223dcd77e03d473db0f6ee,/home/lab1006/Desktop/hdfile1.txt offset 0-7
8,b71defb1723cd42ba910645bcd13b206,/home/lab1006/Desktop/hdfile1.txt offset 8-15
8,4ccec7f35788f66ffc0a056743a5ae2a,/home/lab1006/Desktop/hdfile1.txt offset 16-23
8,184b35e1469d0cbc412449dd3c863453,/home/lab1006/Desktop/hdfile1.txt offset 24-31
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep c md5 -r /home/lab1006/Desktop/hashfile
/home/lab1006/Desktop/c: No such file or directory
/home/lab1006/Desktop/md5: No such file or directory
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -r c md5 /home/lab1006/Desktop/hashfile
##
67,569c461fe9b706369b7617ece7567391,6a1b121b9890783bc614af2d6851d84476ddd91808e3a368a50b5622885e48e2,/home/lab1006/Desktop/hashfile/hdfile2.tx
t
```

```
Activities  Terminal  Wed 12:17
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,hdfile1
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s hdfile1
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -s hdfile1
##
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,/home/lab1006/Desktop/hdfile1
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5,sha1,sha256,tiger hdfile1
%%%% HASHDEEP-1.0
%%%% size,md5,sha1,sha256,tiger,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5,sha1,sha256,tiger hdfile1
##
32,3592e22887f8f7fc071f7a981b28534e,30c13a60bdd015bd44da4264f17d92917b0a8736,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,
b81ae4ea62cfbab32c295baf841dc11acc4cf758734043f0,/home/lab1006/Desktop/hdfile1
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5 *.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5 result.txt
##
384,ab72a33ce65e9bef6be12decf9e0a4d2,/home/lab1006/Desktop/result.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5 *.txt
%%%% HASHDEEP-1.0
%%%% size,md5,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5 hdfile1.txt result.txt
##
384,ab72a33ce65e9bef6be12decf9e0a4d2,/home/lab1006/Desktop/result.txt
32,3592e22887f8f7fc071f7a981b28534e,/home/lab1006/Desktop/hdfile1.txt
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -c md5,sha1 *.txt
%%%% HASHDEEP-1.0
%%%% size,md5,sha1,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -c md5,sha1 hdfile1.txt result.txt
##
32,3592e22887f8f7fc071f7a981b28534e,30c13a60bdd015bd44da4264f17d92917b0a8736,/home/lab1006/Desktop/hdfile1.txt
```

```
Activities  Terminal  Wed 12:17
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man hashdeep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man wifite
No manual entry for wifite
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c
hashdeep: option requires an argument -- 'c'
Try `hashdeep -h` for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c <>
bash: syntax error near unexpected token `newline'
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c md5 sha256
/home/lab1006/sha256: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -c <md5,sha256>
bash: syntax error near unexpected token `newline'
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man hashdeep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man md5deep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man md5
No manual entry for md5
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man hashdeep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ man md5deep
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ cd desktop
bash: cd: desktop: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ cd home
bash: cd: home: No such file or directory
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ cd Desktop
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep hdfile1
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep hdfile1
##
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,/home/lab1006/Desktop/hdfile1
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -b hdfile1
%%%% HASHDEEP-1.0
%%%% size,md5,sha256,filename
## Invoked from: /home/lab1006/Desktop
## $ hashdeep -b hdfile1
##
32,3592e22887f8f7fc071f7a981b28534e,6027b1c6aa0203fbcb169dd521241b48cc9dbb564e68a9615745b7118bcc9e80,hdfile1
lab1006@lab1006-HP-280-G4-MT-Business-PC:~/Desktop$ hashdeep -s hdfile1
```

```
Activities  Terminal  Wed 12:17
root@lab1006-HP-280-G4-MT-Business-PC: /home/lab1006/Desktop
File  Edit  View  Search  Terminal  Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo apt-get install hashdeep
[sudo] password for lab1006:
Reading package lists... Done
Building dependency tree
Reading state information... Done
hashdeep is already the newest version (4.4-4).
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-125 linux-hwe-5.4-headers-5.4.0-126
  linux-hwe-5.4-headers-5.4.0-128 linux-hwe-5.4-headers-5.4.0-131
  linux-hwe-5.4-headers-5.4.0-135 linux-hwe-5.4-headers-5.4.0-136
  linux-hwe-5.4-headers-5.4.0-137 linux-hwe-5.4-headers-5.4.0-144
  linux-hwe-5.4-headers-5.4.0-146 linux-hwe-5.4-headers-5.4.0-147
  linux-hwe-5.4-headers-5.4.0-148
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 253 not upgraded.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -V
4.4
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ hashdeep -h
hashdeep version 4.4 by Jesse Kornblum and Simson Garfinkel.
$ hashdeep [OPTION]... [FILES]...
-c <alg1,[alg2]> - Compute hashes only. Defaults are MD5 and SHA-256
                   legal values: md5,sha1,sha256,tiger,whirlpool,
-p <size> - piecewise mode. Files are broken into blocks for hashing
-r        - recursive mode. All subdirectories are traversed
-d        - output in DFXML (Digital Forensics XML)
-k <file> - add a file of known hashes
-a        - audit mode. Validates FILES against known hashes. Requires -k
-m        - matching mode. Requires -k
-x        - negative matching mode. Requires -k
-w        - in -m mode, displays which known file was matched
-M and -X act like -m and -x, but display hashes of matching files
-e        - compute estimated time remaining for each file
-s        - silent mode. Suppress all error messages
-b        - prints only the bare name of files; all path information is omitted
-l        - print relative paths for filenames
-i/-I     - only process files smaller than the given threshold
-o        - only process certain types of files. See README/manpage
-v        - verbose mode. Use again to be more verbose
```

Conclusion :

Studied and explored more about hashdeep tool to hash user defined files using several hashing algorithms such as md5 and sha-256 . Also used hashdeep tool to generate , match and audit hash files .

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 02/08/2023

Lab Assignment No:-6

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

Lab Outcome Attained :- LO3

Theory:-

What is the important information that attackers look for using whois command and what attacks can be performed using this information?

The whois command is a powerful tool that can be used **to gather information about domain names, IP addresses, and network devices.** This information can be valuable to attackers, who can use it to perform a **variety of attacks,** including:

Social engineering attacks: Attackers can use the registrant information from a whois lookup to target individuals or organizations with social engineering attacks. For example, they might send phishing emails or phone calls pretending to be from the registrant, in an attempt to trick the victim into revealing sensitive information.
DNS attacks: Attackers can use the name servers listed in a whois lookup to perform DNS attacks, such as DNS spoofing or DNS cache poisoning. These attacks can redirect traffic to malicious websites or servers, or they can disrupt legitimate websites and services.
IP address tracking: Attackers can use the IP addresses listed in a whois lookup to track the physical location of a domain or website. This information can be used to launch targeted attacks, or to gather intelligence about a target organization.

Fraudulent domain registration: Attackers can use the whois command to register domains that are similar to the names of legitimate businesses or organizations. This can be used to trick users into visiting malicious websites or to impersonate the legitimate business or organization.

How traceroute command works in order to trace the route of given host?

The traceroute command works by sending Internet Control Message Protocol **(ICMP) packets** to the destination host, with the **Time to Live (TTL)** field set to a low value. The TTL field is used to determine how long a packet can exist in a network before it is discarded.

When a router receives an ICMP packet with a TTL of 1, it decrements the TTL value by 1 and then sends the packet to the next hop in the route to the destination host. If the TTL value reaches 0 before the packet reaches the destination host, the router will send an ICMP **"Time Exceeded" message back to the source host.**

The traceroute command sends a series of ICMP packets with increasing TTL values. For each packet, the traceroute command will record the IP address of the router that sent the "Time Exceeded" message. This information is used to build a list of the hops that the packet took to reach the destination host.

The traceroute command can also be used to measure the **round-trip time (RTT)** for each hop. The RTT is the time it takes for a packet to travel from the source host to the destination host and back again. The RTT can be used to identify potential bottlenecks in the network.

Explain dig command with various options.

The dig command is a tool for **querying Domain Name System (DNS) servers.** It can be used to **lookup the IP address of a domain name, the hostname of an IP address, and other DNS records.**

The dig command has a number of options that can be used to customize the query. Some of the most common options include:

-t: The type of DNS record to query. The most common types are A (IP address), MX (mail exchange), NS (name server), and SOA (start of authority).
-c: The class of DNS record to query. The most common class is IN (Internet).

-q: The domain name or IP address to query.
-x: The hexadecimal value of the query type. This option is used to query for non-standard DNS record types.
-p: The port number of the DNS server to query. The default port number is 53.
-v: Verbose output. This option will print additional information about the query, such as the time it took to complete and the name servers that were queried.

example of how the dig command can be used to lookup the IP address of the Google homepage:

dig google.com
This command will **query the DNS server for the IP address of the Google homepage. The output of the command will show the IP address of the Google homepage, along with other information about the query, such as the time it took to complete and the name servers that were queried.**

Explain any two vulnerabilities detected for the website that you have scanned using nikto. Which attacks are possible if these vulnerabilities are exploited?

vulnerabilities that can be detected for a website using Nikto:

Directory indexing: Directory indexing is a security vulnerability that allows unauthorized users to view the contents of a directory on a web server. This can be exploited by attackers to find sensitive files, such as configuration files or source code.
Outdated software: Outdated software is a security vulnerability that can allow attackers to exploit known vulnerabilities in the software. This is a common vulnerability, as many organizations do not keep their software up to date.

If these vulnerabilities are exploited, attackers can perform a **variety of attacks,** including:

File disclosure: Attackers can use directory indexing to view the contents of a directory on a web server, including sensitive files such as configuration files or source code. This information can be used to launch other attacks, such as gaining access to the web server or stealing sensitive data.
Remote code execution: Attackers can exploit outdated software to execute arbitrary code on a web server. This allows attackers to take control of the web server and perform any action they want, such as stealing data, installing malware, or launching denial-of-service attacks.

Cross-site scripting (XSS): Attackers can exploit outdated software to inject malicious code into a web page. This code can then be executed by unsuspecting users, who may be tricked into providing sensitive information or clicking on malicious links.

Write commands for email harvesting and subdomain harvesting.

theHarvester is a neat information-gathering tool used by both ethical and non-ethical hackers to scrape up emails, subdomains, hosts, employee names, open ports, and banners from different public sources like popular search engines, PGP key servers, and the Shodan database.

Eg:-
theharvester -d microsoft.com -b pgp, searches for e-mail accounts for the domain microsoft.com in a PGP server

theharvester -d theguardian.com -b pgp.
This command will tell theHarvester to search for email accounts with the domain name "theguardian.com" in a pgp server, which is used for encrypting emails.

Other options in theharvestor command are :-
options:
-h, --help
show this help message and exit
-d DOMAIN, --domain DOMAIN
Company name or domain to search.
-l LIMIT, --limit LIMIT
Limit the number of search results, default=500.
-S START, --start START
Start with result number X, default=0.
-p, --proxies
Use proxies for requests, enter proxies in
proxies.yaml.
-s, --shodan
Use Shodan to query discovered hosts.
--screenshot SCREENSHOT
Take screenshots of resolved domains specify output
directory: --screenshot output_directory
-v, --virtual-host
Verify host name via DNS resolution and search for
virtual hosts.
-e DNS_SERVER, --dns-server DNS_SERVER

DNS server to use for lookup.

-t, --take-over

Check for takeovers.

-r [DNS_RESOLVE], --dns-resolve [DNS_RESOLVE]

Perform DNS resolution on subdomains with a resolver list or passed in resolvers, default False.

-n, --dns-lookup

Enable DNS server lookup, default False.

-c, --dns-brute

Perform a DNS brute force on the domain.

-f FILENAME, --filename FILENAME

Save the results to an XML and JSON file.

-b SOURCE, --source SOURCE

anubis, baidu, bevigil, binaryedge, bing, bingapi, bufferoverun, brave, censys, certspotter, criminalip, crtsh, dnsdumpster, duckduckgo, fullhunt, github-code, hackertarget, hunter, hunterhow, intelx, otx, pentesttools, projectdiscovery, rapiddns, rocketreach, securityTrails, sitedossier, subdomainfinderc99, threatminer, urlscan, virustotal, yahoo, zoomeye

What are different functionalities provided by dimtry. Write Dmitry command for whois lookup, an IP whois lookup, retrieve Netcraft info, search for subdomains, search for email addresses, do a TCP port scan, and save the output to example.txt for the domain example.com

The dmitry command in Linux is a command-line tool that can be used to gather information about a computer's hardware. It can be used to identify the computer's make and model, the processor type, the amount of memory, the hard drive size, and other information.

The dmitry command is a powerful tool that can be used to troubleshoot hardware problems. For example, if you are having problems with your computer's memory, you can use dmitry to identify the type of memory that your computer uses. This information can then be used to purchase the correct replacement memory.

Whois lookup :-
dmitry -i example.txt example.com

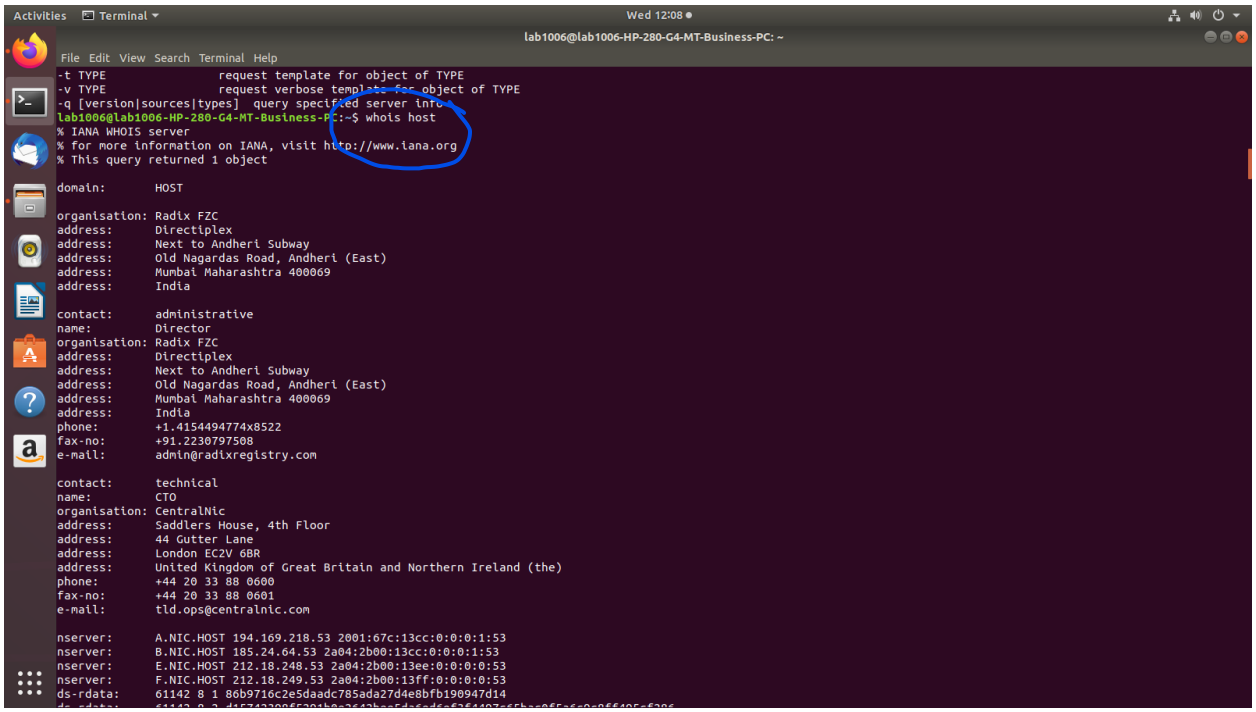
Retrieve netcraft info :-
dmitry -n example.txt example.com

Search for subdomain :-
dmitry -s example.txt example.com

Search for email addresses :-
dmitry -e example.txt example.com

Do a TCP port scan :-
dmitry -p example.txt example.com

Output Screenshot:-



```
organisation: Radix FZC
address:      Directiplex
address:      Next to Andheri Subway
address:      Old Nagardas Road, Andheri (East)
address:      Mumbai Maharashtra 400069
address:      India

contact:      administrative
name:         Director
organisation: Radix FZC
address:      Directiplex
address:      Next to Andheri Subway
address:      Old Nagardas Road, Andheri (East)
address:      Mumbai Maharashtra 400069
address:      India
phone:        +1.4154494774x8522
fax-no:       +91.2230797508
e-mail:       admin@radixregistry.com

contact:      technical
name:         CTO
organisation: CentralNic
address:      Saddlers House, 4th Floor
address:      44 Gutter Lane
address:      London EC2V 6BR
address:      United Kingdom of Great Britain and Northern Ireland (the)
phone:        +44 20 33 88 0600
fax-no:       +44 20 33 88 0601
e-mail:       tld.ops@centralnic.com

nserver:      A.NIC.HOST 194.169.218.53 2001:67c:13cc:0:0:0:1:53
nserver:      B.NIC.HOST 185.24.64.53 2a04:2b00:13cc:0:0:0:1:53
nserver:      E.NIC.HOST 212.18.248.53 2a04:2b00:13ee:0:0:0:0:53
nserver:      F.NIC.HOST 212.18.249.53 2a04:2b00:13ff:0:0:0:0:53
ds-rdata:     61142 8 1 86b9716c2e5daadc785ada27d4e8bfb190947d14
ds-rdata:     61142 8 2 d15742398f5291b0e2642bee5da6ed6ef3f4497c65bac0f5a6c9c8ff495cf286

whois:        whois.nic.host

status:       ACTIVE
remarks:      Registration information: http://radixregistry.com/

created:      2014-05-22
changed:      2021-10-12
source:       IANA
```

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ whois tsec.edu
This Registry database contains ONLY .EDU domains.
The data in the EDUCAUSE Whois database is provided
by EDUCAUSE for information purposes in order to
assist in the process of obtaining information about
or related to .edu domain registration records.

The EDUCAUSE Whois database is authoritative for the
.EDU domain.

A Web interface for the .EDU EDUCAUSE Whois Server is
available at: http://whois.educause.edu

By submitting a Whois query, you agree that this information
will not be used to allow, enable, or otherwise support
the transmission of unsolicited commercial advertising or
solicitations via e-mail.  The use of electronic processes to
harvest information from this server is generally prohibited
except as reasonably necessary to register or modify .edu
domain names.

-----------------------------------------------------------

Domain Name: TSEC.EDU

Registrant:
        Thadomal Sahani Engineering College
        P.G Kher Marg, Bandra(W)
        Mumbai, Maharashtra 400 050
        India

Administrative Contact:
        Dr. Gopakumaran Thampi
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        gtthampi@yahoo.com

Technical Contact:
        Chetan Agarwal
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
```

```
Technical Contact:
        Chetan Agarwal
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        chetan.agarwal@thadomal.org

Name Servers:
        NS2.SALESUPP.IN
        NS1.SALESUPP.IN

Domain record activated:    22-Jan-2001
Domain record last updated: 01-Aug-2023
Domain expires:             31-Jul-2023
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ whois ibm.com
   Domain Name: IBM.COM
   Registry Domain ID: 1555443_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.corporatedomains.com
   Registrar URL: http://cscdbs.com
   Updated Date: 2023-03-16T05:11:31Z
   Creation Date: 1986-03-19T05:00:00Z
   Registry Expiry Date: 2024-03-20T04:00:00Z
   Registrar: CSC Corporate Domains, Inc.
   Registrar IANA ID: 299
   Registrar Abuse Contact Email: domainabuse@cscglobal.com
   Registrar Abuse Contact Phone: 8887802723
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   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: ASIA3.AKAM.NET
   Name Server: EUR2.AKAM.NET
   Name Server: EUR5.AKAM.NET
   Name Server: NS1-206.AKAM.NET
   Name Server: NS1-99.AKAM.NET
   Name Server: USC2.AKAM.NET
   Name Server: USC3.AKAM.NET
   Name Server: USW2.AKAM.NET
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2023-08-02T05:32:02Z <<<

For more information on Whois status codes, please visit https://icann.org/epp
```

```
#
# If you see inaccuracies in the results, please report at
# https://www.arin.net/resources/registry/whois/inaccuracy_reporting/
#
# Copyright 1997-2023, American Registry for Internet Numbers, Ltd.
#

lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ traceroute tsec.edu
traceroute to tsec.edu (162.241.70.62), 64 hops max
  1   192.168.0.1  2.564ms  3.855ms  4.194ms
  2   203.212.25.1  6.631ms  6.101ms  4.850ms
  3   203.212.24.53  3.552ms  6.164ms  11.148ms
  4   175.100.177.53  21.012ms  6.548ms  5.061ms
  5   172.16.2.101  7.187ms  8.798ms  10.221ms
  6   121.241.43.57  9.018ms  10.463ms  8.800ms
  7   172.23.78.237  10.059ms  9.615ms  9.882ms
  8   180.87.38.5  8.902ms  10.816ms  9.147ms
  9   *  *  *
 10   *  *  *
 11   80.231.153.168  139.926ms  *  *
 12   80.231.153.21  134.060ms  *  *
 13   50.6.131.2  248.286ms  211.832ms  205.097ms
 14   *  *  *
 15   162.241.70.62  208.098ms !*  207.521ms !*  207.574ms !*
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ dig tsec.edu

; <<>> DiG 9.11.3-1ubuntu1.18-Ubuntu <<>> tsec.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 8973
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;tsec.edu.                      IN      A

;; ANSWER SECTION:
tsec.edu.               5476    IN      A       162.241.70.62

;; Query time: 0 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Wed Aug 02 11:18:01 IST 2023
;; MSG SIZE  rcvd: 53
```

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

Non-authoritative answer:
Name:   tsec.edu
Address: 162.241.70.62

lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ nikto -h tsec.edu
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          162.241.70.62
+ Target Hostname:    tsec.edu
+ Target Port:        80
+ Start Time:         2023-08-02 11:34:30 (GMT5.5)
---------------------------------------------------------------------------
+ Server: Apache
+ The anti-clickjacking X-Frame-Options header is not present.
+ Root page / redirects to: https://tsec.edu/
^[[B^[[C^Clab1006@lab1006-HP-280-G4-MT-Business-PC:~$ traceroute -v tsec.edu
traceroute: invalid option -- 'v'
Try 'traceroute --help' or 'traceroute --usage' for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ traceroute -v 162.241.70.62
traceroute: invalid option -- 'v'
Try 'traceroute --help' or 'traceroute --usage' for more information.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ traceroute verbose 162.241.70.62
traceroute to 162.241.70.62 (162.241.70.62), 64 hops max
  1   192.168.0.1  0.918ms  0.800ms  0.778ms
  2   203.212.25.1  1.628ms  1.151ms  1.184ms
  3   203.212.24.53  1.448ms  1.194ms  1.011ms
  4   175.100.177.53  2.993ms  2.309ms  1.974ms
  5   172.16.2.101  5.893ms  3.071ms  2.828ms
  6   121.241.43.57  3.380ms  *  *
  7   *  172.23.78.237  3.017ms  12.435ms
  8   180.87.38.5  3.291ms  2.782ms  3.176ms
  9   195.219.174.16  117.306ms  116.898ms  *
 10   195.219.174.9  126.179ms  *  125.635ms
 11   *  *  *
 12   *  *  *
 13   50.6.131.2  214.564ms  215.921ms  213.818ms
 14   *  *  *
 15   162.241.70.62  207.589ms !*  207.424ms !*  208.432ms !*
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ nikto -h tsec.edu
- Nikto v2.1.5
---------------------------------------------------------------------------
```

```
*Requires the -p flagged to be passed
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ dmitry tsec.edu
Deepmagic Information Gathering Tool
"There be some deep magic going on"

HostIP:162.241.70.62
HostName:tsec.edu

Gathered Inet-whois information for 162.241.70.62
---------------------------------

inetnum:        162.222.91.0 - 162.244.51.255
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:        2019-01-07T10:49:27Z
last-modified:  2019-01-07T10:49:27Z
source:         RIPE
```

```
last-modified:  2001-09-22T09:31:27Z
source:         RIPE # Filtered

% This query was served by the RIPE Database Query Service version 1.107 (DEXTER)



Gathered Inic-whois information for tsec.edu
---------------------------------

Domain Name: TSEC.EDU

Registrant:
        Thadomal Sahani Engineering College
        P.G Kher Marg, Bandra(W)
        Mumbai, Maharashtra 400 050
        India

Administrative Contact:
        Dr. Gopakumaran Thampi
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        gtthampi@yahoo.com

Technical Contact:
        Chetan Agarwal
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        chetan.agarwal@thadomal.org

Name Servers:
        NS2.SALESUPP.IN
        NS1.SALESUPP.IN

Domain record activated:    22-Jan-2001
Domain record last updated: 01-Aug-2023
Domain expires:             31-Jul-2023

Gathered Netcraft information for tsec.edu
```

```
---------------------------------

Retrieving Netcraft.com information for tsec.edu
Netcraft.com Information gathered

Gathered Subdomain information for tsec.edu
---------------------------------
Searching Google.com:80...
HostName:www.tsec.edu
HostIP:162.241.70.62
HostName:alumni.tsec.edu
HostIP:13.213.42.252
Searching Altavista.com:80...
Found 2 possible subdomain(s) for host tsec.edu, Searched 0 pages containing 0 results

Gathered E-Mail information for tsec.edu
---------------------------------
Searching Google.com:80...
Searching Altavista.com:80...
Found 0 E-Mail(s) for host tsec.edu, Searched 0 pages containing 0 results

Gathered TCP Port information for 162.241.70.62
---------------------------------

 Port           State

22/tcp          open
25/tcp          open
53/tcp          open
80/tcp          open
110/tcp         open
143/tcp         open

Portscan Finished: Scanned 150 ports, 99 ports were in state closed


All scans completed, exiting
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ theharvestor -d tsec.edu
theharvestor: command not found
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo apt install theharvestor
[sudo] password for lab1006:
E: dpkg was interrupted, you must manually run 'sudo dpkg --configure -a' to correct the problem.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo apt install theharvestor
E: dpkg was interrupted, you must manually run 'sudo dpkg --configure -a' to correct the problem.
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
```

Conclusion:-

Studied various network reconnaissance tools such as WHOIS , dig , traceroute , nslookup , nikto , dmitry and explored various options related to these tools , gathered information and insights about network and domain registrars

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 23/08/2023

Lab Assignment No:-7

Aim:-Study of packet sniffer tools TCPDUMP.

Lab Outcome Attained :- LO3

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'

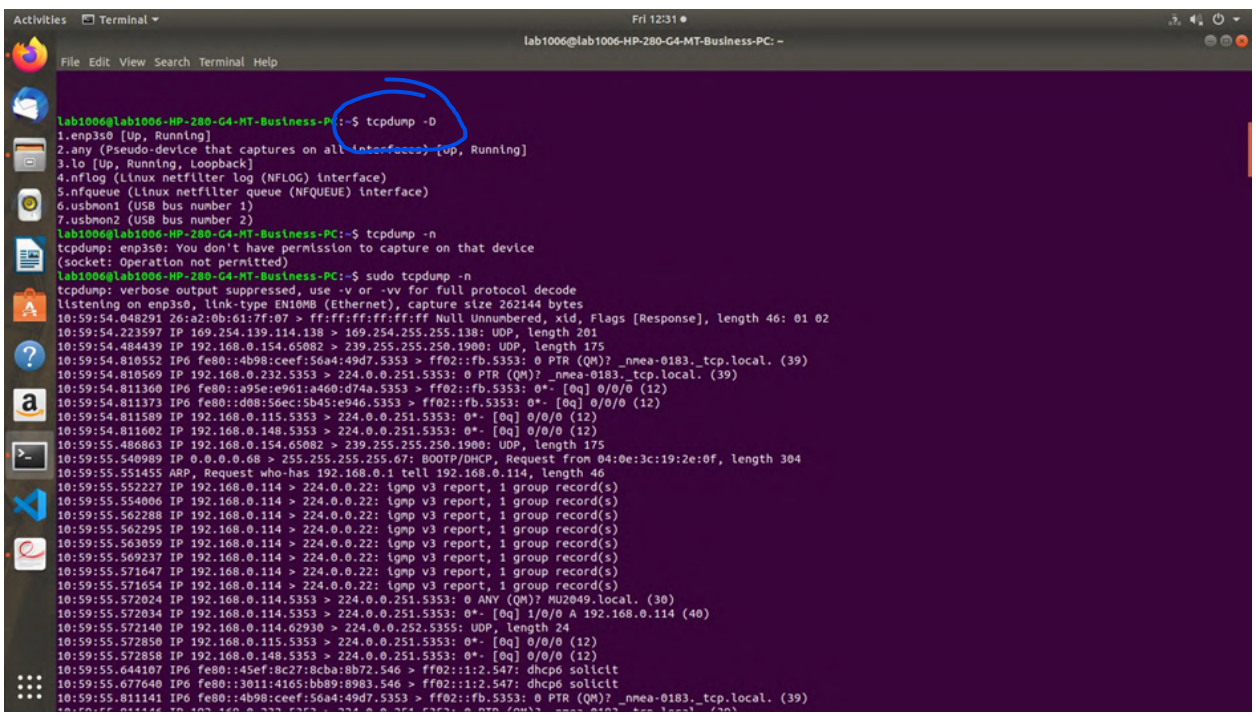
This captures only SYN packets (TCP packets with the SYN flag set).

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 Screenshot:-



```
Activities  Terminal                Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
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 'ssdp' 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-ti
me 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-ser
ver 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)
```

```
Activities  Terminal                Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
    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:e946.53
53 > 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 192.168.0.1
85, 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 192.168.0.1
90, 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 192.168.0.1
90, 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], leng
th 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:
```

```
Activities  Terminal                                  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
    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 192.168.0.1
90, 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 192.168.0.1
85, 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 e
cr 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 355497141
6], 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 309342
6561], 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 242720
383], 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 130
5689667], 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}], leng
th 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
```

```
Activities  Terminal                                  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
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 e
cr 1089564564], length 0
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
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 1
089565016 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 91.189.9
1.48, 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 2
620:2d: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:f9
80:93a1, 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)
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
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)
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
12:04:44.335645 IP ip98.ip-51-75-86.eu.https > lab1006-HP-280-G4-MT-Business-PC.42950: Flags [R], seq 3863433480, win 0, length 0
12:04:44.335649 IP ip98.ip-51-75-86.eu.https > lab1006-HP-280-G4-MT-Business-PC.42950: Flags [R], seq 3863433480, win 0, length 0
12:04:44.335649 IP ip98.ip-51-75-86.eu.https > lab1006-HP-280-G4-MT-Business-PC.42950: Flags [R], seq 3863433480, win 0, length 0
12:04:55.146342 IP 39.12.213.35.bc.googleusercontent.com.https > lab1006-HP-280-G4-MT-Business-PC.48000: Flags [R], seq 585098989, win 0, length 0
12:04:55.146361 IP 39.12.213.35.bc.googleusercontent.com.https > lab1006-HP-280-G4-MT-Business-PC.48000: Flags [R], seq 585098989, win 0, length 0
^C
5 packets captured
5 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump 'tcp[13] & 1!=0'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:05:20.015253 IP 39.12.213.35.bc.googleusercontent.com.https > lab1006-HP-280-G4-MT-Business-PC.48012: Flags [F.], seq 2629149024, ack 1929302308, win 501, options [n
op,nop,TS val 2466317305 ecr 2922664599], length 0
12:05:20.015507 IP lab1006-HP-280-G4-MT-Business-PC.48012 > 39.12.213.35.bc.googleusercontent.com.https: Flags [F.], seq 32, ack 1, win 501, options [nop,nop,TS val 292
2729599 ecr 2466317305], length 0
12:05:21.308781 IP lab1006-HP-280-G4-MT-Business-PC.43518 > bom12s13-in-f10.1e100.net.https: Flags [F.], seq 2428652434, ack 1126368455, win 501, options [nop,nop,TS va
l 2874683512 ecr 3493271097], length 0
12:05:21.310519 IP bom12s13-in-f10.1e100.net.https > lab1006-HP-280-G4-MT-Business-PC.43518: Flags [F.], seq 1, ack 0, win 267, options [nop,nop,TS val 3493271099 ecr 2
874683512], length 0
12:05:31.935100 IP lab1006-HP-280-G4-MT-Business-PC.34760 > bom07s36-in-f2.1e100.net.https: Flags [F.], seq 1180428611, ack 3813265531, win 501, options [nop,nop,TS val
 41552518 ecr 1543554862], length 0
12:05:31.937062 IP bom07s36-in-f2.1e100.net.https > lab1006-HP-280-G4-MT-Business-PC.34760: Flags [F.], seq 1, ack 0, win 265, options [nop,nop,TS val 1543554864 ecr 41
552518], length 0
12:05:36.868948 IP lab1006-HP-280-G4-MT-Business-PC.50560 > 103.226.190.44.https: Flags [F.], seq 1711529759, ack 2298162122, win 501, options [nop,nop,TS val 319482289
2 ecr 583854437], length 0
12:05:36.871338 IP 103.226.190.44.https > lab1006-HP-280-G4-MT-Business-PC.50560: Flags [F.], seq 1, ack 0, win 261, options [nop,nop,TS val 583859434 ecr 3194822892],
length 0
12:05:43.871629 IP lab1006-HP-280-G4-MT-Business-PC.44260 > ec2-44-215-138-223.compute-1.amazonaws.com.https: Flags [F.], seq 3141369856, ack 2220810018, win 501, optio
ns [nop,nop,TS val 1678878368 ecr 2067633469], length 0
12:05:44.060653 IP ec2-44-215-138-223.compute-1.amazonaws.com.https > lab1006-HP-280-G4-MT-Business-PC.44260: Flags [F.], seq 1, ack 0, win 479, options [nop,nop,TS val
 2067636189 ecr 1678878368], length 0
12:05:45.873434 IP lab1006-HP-280-G4-MT-Business-PC.43688 > 52.46.151.131.https: Flags [F.], seq 400986082, ack 208373217, win 501, length 0
12:05:46.068962 IP 52.46.151.131.https > lab1006-HP-280-G4-MT-Business-PC.43688: Flags [F.], seq 1, ack 0, win 942, 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 'tcp[tcpflags] == tcp-rst'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:09:45.018984 IP lab1006-HP-280-G4-MT-Business-PC.48656 > 102.115.120.34.bc.googleusercontent.com.https: Flags [R], seq 2984745258, win 0, length 0
12:09:45.019042 IP lab1006-HP-280-G4-MT-Business-PC.48656 > 102.115.120.34.bc.googleusercontent.com.https: Flags [R], seq 2984745258, win 0, length 0
12:09:51.570479 IP lab1006-HP-280-G4-MT-Business-PC.36552 > 104.17.25.14.https: Flags [R], seq 1050894372, win 0, length 0
12:09:51.578512 IP lab1006-HP-280-G4-MT-Business-PC.36532 > 104.17.25.14.https: Flags [R], seq 1460208463, win 0, length 0
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
12 packets captured
12 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump 'tcp[tcpflags] == tcp-rst'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:09:45.018984 IP lab1006-HP-280-G4-MT-Business-PC.48656 > 102.115.120.34.bc.googleusercontent.com.https: Flags [R], seq 2984745258, win 0, length 0
12:09:45.019042 IP lab1006-HP-280-G4-MT-Business-PC.48656 > 102.115.120.34.bc.googleusercontent.com.https: Flags [R], seq 2984745258, win 0, length 0
12:09:51.570479 IP lab1006-HP-280-G4-MT-Business-PC.36552 > 104.17.25.14.https: Flags [R], seq 1050894372, win 0, length 0
12:09:51.578512 IP lab1006-HP-280-G4-MT-Business-PC.36532 > 104.17.25.14.https: Flags [R], seq 1460208463, win 0, length 0
12:09:51.581245 IP lab1006-HP-280-G4-MT-Business-PC.36532 > 104.17.25.14.https: Flags [R], seq 1460208487, win 0, length 0
12:09:51.581249 IP lab1006-HP-280-G4-MT-Business-PC.36532 > 104.17.25.14.https: Flags [R], seq 1460208488, win 0, length 0
12:09:52.167861 IP lab1006-HP-280-G4-MT-Business-PC.56434 > bom07s32-in-f3.1e100.net.https: Flags [R], seq 3656444749, win 0, length 0
12:09:52.167868 IP lab1006-HP-280-G4-MT-Business-PC.56444 > bom07s32-in-f3.1e100.net.https: Flags [R], seq 2520567994, win 0, length 0
12:09:52.997559 IP lab1006-HP-280-G4-MT-Business-PC.50482 > bom07s36-in-f6.1e100.net.https: Flags [R], seq 207038670, win 0, length 0
12:09:52.997636 IP lab1006-HP-280-G4-MT-Business-PC.50482 > bom07s36-in-f6.1e100.net.https: Flags [R], seq 207038670, win 0, length 0
12:09:52.997648 IP lab1006-HP-280-G4-MT-Business-PC.50482 > bom07s36-in-f6.1e100.net.https: Flags [R], seq 207038670, win 0, length 0
12:09:52.997649 IP lab1006-HP-280-G4-MT-Business-PC.50482 > bom07s36-in-f6.1e100.net.https: Flags [R], seq 207038670, win 0, length 0
12:09:58.330850 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.330930 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.331079 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.331146 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.331651 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.331663 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.331763 IP lab1006-HP-280-G4-MT-Business-PC.41382 > venuepool.venuepool.com.https: Flags [R], seq 2568885250, win 0, length 0
12:09:58.518067 IP lab1006-HP-280-G4-MT-Business-PC.41370 > venuepool.venuepool.com.https: Flags [R], seq 2400063489, win 0, length 0
12:09:58.518141 IP lab1006-HP-280-G4-MT-Business-PC.41370 > venuepool.venuepool.com.https: Flags [R], seq 2400063489, win 0, length 0
12:09:58.518147 IP lab1006-HP-280-G4-MT-Business-PC.41370 > venuepool.venuepool.com.https: Flags [R], seq 2400063489, win 0, length 0
12:09:58.518164 IP lab1006-HP-280-G4-MT-Business-PC.41370 > venuepool.venuepool.com.https: Flags [R], seq 2400063489, win 0, length 0
12:10:11.001618 IP lab1006-HP-280-G4-MT-Business-PC.60294 > 151.101.153.229.https: Flags [R], seq 3683966171, win 0, length 0
12:10:11.001840 IP lab1006-HP-280-G4-MT-Business-PC.60294 > 151.101.153.229.https: Flags [R], seq 3683966195, win 0, length 0
12:10:11.001890 IP lab1006-HP-280-G4-MT-Business-PC.60294 > 151.101.153.229.https: Flags [R], seq 3683966195, win 0, length 0
12:10:11.001903 IP lab1006-HP-280-G4-MT-Business-PC.60294 > 151.101.153.229.https: Flags [R], seq 3683966195, win 0, length 0
12:10:11.002174 IP lab1006-HP-280-G4-MT-Business-PC.60294 > 151.101.153.229.https: Flags [R], seq 3683966196, win 0, length 0
12:10:11.390956 IP lab1006-HP-280-G4-MT-Business-PC.55150 > bom07s32-in-f3.1e100.net.https: Flags [R], seq 1738671314, win 0, length 0
12:10:11.391016 IP lab1006-HP-280-G4-MT-Business-PC.55150 > bom07s32-in-f3.1e100.net.https: Flags [R], seq 1738671314, win 0, length 0
12:10:33.454979 IP lab1006-HP-280-G4-MT-Business-PC.55754 > bom12s13-in-f22.1e100.net.https: Flags [R], seq 3496861439, win 0, length 0
12:10:33.455808 IP lab1006-HP-280-G4-MT-Business-PC.55754 > bom12s13-in-f22.1e100.net.https: Flags [R], seq 3496861463, win 0, length 0
12:10:33.455809 IP lab1006-HP-280-G4-MT-Business-PC.55754 > bom12s13-in-f22.1e100.net.https: Flags [R], seq 3496861464, win 0, length 0
12:10:38.236441 IP lab1006-HP-280-G4-MT-Business-PC.40492 > 151.101.153.229.https: Flags [R], seq 2267406728, win 0, length 0
12:10:38.236517 IP lab1006-HP-280-G4-MT-Business-PC.40492 > 151.101.153.229.https: Flags [R], seq 2267406728, win 0, length 0
12:10:38.236519 IP lab1006-HP-280-G4-MT-Business-PC.40492 > 151.101.153.229.https: Flags [R], seq 2267406729, win 0, length 0
12:10:40.533314 IP lab1006-HP-280-G4-MT-Business-PC.60308 > 151.101.153.229.https: Flags [R], seq 2608403810, win 0, length 0
12:10:40.533465 IP lab1006-HP-280-G4-MT-Business-PC.60308 > 151.101.153.229.https: Flags [R], seq 2608403810, win 0, length 0
12:10:40.534225 IP lab1006-HP-280-G4-MT-Business-PC.60308 > 151.101.153.229.https: Flags [R], seq 2608403811, win 0, length 0
```

Conclusion:-

Learnt about how TCPDump can be used in practical life and how can it be used to capture , dissect , interpret network packets , offering various insights about network behavior , troubleshooting and security assessment . Also explored various commands related to TCPDump

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 13/09/2023

Lab Assignment No:-8

Aim:- 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.

Lab Outcome Attained :- LO4

Theory:-

1. What is port scanning ? What is Nmap?

Port scanning is a crucial technique in the realm of networking and cybersecurity. It involves probing a host or network to discover open ports, which act as gateways for network services or applications. These ports are the entry points through which data flows in and out of a system.

Each port is associated with a specific protocol and service, making them essential for communication within a network. The primary purpose of port scanning is to unveil the network landscape, assess the security posture of a system, and identify potential vulnerabilities or attack vectors. Network administrators use it for troubleshooting and monitoring network health, while security professionals use it to detect and mitigate security threats.

However, malicious actors also leverage port scanning to identify vulnerable targets for cyberattacks.Nmap, short for "Network Mapper," is a versatile and widely used open-source tool for network discovery and security auditing. Developed by Gordon Lyon, also known as Fyodor, Nmap has earned a reputation as the go-to tool for port scanning due to its comprehensive feature set and cross-platform compatibility. Nmap's capabilities extend beyond basic port scanning.

It can perform a wide range of tasks, including host discovery, service and version detection, operating system fingerprinting, and scripting for automation. Nmap is favored by security professionals, network administrators, and ethical hackers for its flexibility and accuracy

2. Explain in brief different states of port.

Ports are essential components of network communication, and they can exist in various states, each conveying specific information about their accessibility and functionality. Understanding these port states is crucial for network administrators and security professionals. Here are brief explanations of different port states:

1. Open: An "open" port is one that is actively listening for incoming connections. It indicates that a network service or application is running and ready to accept data or requests. Open ports are crucial for legitimate network communication.

2. Closed: A "closed" port is one that is not actively listening for connections. It means there is no service or application running on that port. Closed ports are safe from unauthorized access, but they still indicate the presence of a host.

3. Filtered: A "filtered" port is one that cannot be determined as open or closed with certainty. This state often occurs when a firewall, intrusion detection system (IDS), or other security measure blocks incoming requests to the port. It makes it challenging to discern the actual status of the port.

4. Unfiltered: An "unfiltered" port is one that is accessible and can be reached, but its status (open or closed) remains undetermined. Unfiltered ports usually indicate that no significant firewall rules are blocking access to the port.

5. Open | Filtered: This state combines characteristics of both open and filtered ports. It suggests that the port is reachable, but the response to a probing request is filtered, possibly by a firewall. It can be challenging to ascertain the exact state of such ports.

6. Closed | Filtered: This state also combines characteristics of both closed and filtered ports. It implies that the port is accessible, but the response is filtered, typically indicating that a firewall is blocking probing attempts. This state can be confusing during port scanning.

Write the commands for following type of port scanning techniques using NMAP, Explain in 4 to 5 lines how each of them works.

Port scanning is an integral part of network reconnaissance, allowing us to discover open ports on target hosts and gain insights into their configuration and potential vulnerabilities. Nmap (Network Mapper) is a versatile tool that offers various scanning techniques to achieve this.

1. TCP Connect Scan
Command: `nmap -sT target`
Description: This scan emulates a full TCP connection attempt to each target port. If a connection is successfully established, the port is considered open. It's the most straightforward scanning method, but it can be easily detected by intrusion detection systems (IDS) and firewalls because it fully completes the TCP handshake.

2. TCP SYN Scan
Command: `nmap -sS target`
Description: The TCP SYN scan, also known as the "half-open" scan, is stealthier than the TCP Connect scan. It sends SYN (Synchronize) packets to target ports and examines their responses. If a port responds with a SYN-ACK (Synchronize-Acknowledgment) packet, it's considered open. If it responds with an RST (Reset) packet, it's considered closed. This scan doesn't complete the full TCP handshake, making it less likely to trigger alarms.

3. FIN Scan
Command: `nmap -sF target`
Description: The FIN scan sends FIN (Finish) packets to target ports. If a port is closed, it should respond with an RST packet. If it's open, it should ignore the FIN

packet. This scan is effective for identifying systems with non-standard TCP stack implementations.

4. Null Scan
Command: `nmap -sN target`
Description: Similar to the FIN scan, the Null scan sends packets with no TCP flags set, making them appear "null." If a port is closed, it should respond with an RST packet. If it's open, it should ignore the packet. This technique is stealthy and can bypass some firewall rules.

5. XMAS Scan
Command: `nmap -sX target`
Description: The XMAS scan sets multiple TCP flags in the packet, making it look like a Christmas tree. If a port is closed, it should respond with an RST packet. If it's open, it should ignore the packet. Like the Null scan, this method can bypass firewall rules.

6. ACK Scan
Command: `nmap -sA target`
Description: The ACK scan sends ACK (Acknowledgment) packets to target ports. It can be used to determine if a firewall is in place; open ports will typically respond with an RST packet, while filtered ports may not respond at all. This scan can identify packet-filtering firewalls.

7. Ping Sweep
Command: `nmap -sn target_range`
Description: Ping sweeping is not a port scanning technique but is often used to identify live hosts before conducting port scans. It sends ICMP echo requests (pings) to a range of IP addresses and identifies responsive hosts, reducing unnecessary scanning on non-responsive
Hosts.

8. Service and Version Detection
Command: `nmap -sV target`

Description: Nmap can probe open ports to identify the services running on them and their versions. This information is crucial for understanding the potential attack surface and vulnerabilities.

9. Port and Port Range Scanning
Command: `nmap -p ports target`
Description: Nmap allows users to specify individual ports or port ranges for scanning, giving flexibility in targeting specific services or performing broad scans of common ports.

10. OS Fingerprinting
Command: `nmap -O target`
Description: Nmap can attempt to identify the operating system running on the target by analyzing network responses and characteristics. This information aids in understanding the target environment.

Output Screenshots:-

FIND IP USING DIG : dig tsec.edu

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sA 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:07 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.016s latency).
All 1000 scanned ports on _gateway (192.168.0.1) are unfiltered
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.59 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sW 192.168.0.1
[sudo] password for lab1006:

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:22 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.030s latency).
All 1000 scanned ports on _gateway (192.168.0.1) are closed
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.62 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sI 192.168.0.1
WARNING: Many people use -Pn w/Idlescan to prevent pings from their true IP.  On the other hand, timing info Nmap gains from pings can allow for faster, more reliable s
cans.

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:23 IST
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 2.06 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sI 192.168.0.1
WARNING: Many people use -Pn w/Idlescan to prevent pings from their true IP.  On the other hand, timing info Nmap gains from pings can allow for faster, more reliable s
cans.

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:23 IST
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 2.03 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sI 192.168.0.1
WARNING: Many people use -Pn w/Idlescan to prevent pings from their true IP.  On the other hand, timing info Nmap gains from pings can allow for faster, more reliable s
cans.

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:23 IST
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 2.03 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
Nmap done: 1 IP address (1 host up) scanned in 9.66 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sF 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:52 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.00057s latency).
Not shown: 998 open|filtered ports
PORT    STATE  SERVICE
139/tcp closed netbios-ssn
445/tcp closed microsoft-ds
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 18.10 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sX 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:02 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.00055s latency).
Not shown: 998 open|filtered ports
PORT    STATE  SERVICE
139/tcp closed netbios-ssn
445/tcp closed microsoft-ds
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 9.58 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sA 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:06 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.016s latency).
All 1000 scanned ports on _gateway (192.168.0.1) are unfiltered
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.59 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sA 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 12:06 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.017s latency).
All 1000 scanned ports on _gateway (192.168.0.1) are unfiltered
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sA 192.168.0.1
```

```
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.39 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sT 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:43 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.0074s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
443/tcp  open  https
1900/tcp open  upnp
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.38 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sN 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:44 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.00054s latency).
Not shown: 998 open|filtered ports
PORT    STATE  SERVICE
139/tcp closed netbios-ssn
445/tcp closed microsoft-ds
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 5.84 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sN 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:48 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.00056s latency).
Not shown: 998 open|filtered ports
PORT    STATE  SERVICE
139/tcp closed netbios-ssn
445/tcp closed microsoft-ds
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 9.66 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sF 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:52 IST
Nmap scan report for _gateway (192.168.0.1)
```

```
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sS 192.168.0.1
[sudo] password for lab1006:

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:42 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.016s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
443/tcp  open  https
1900/tcp open  upnp
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.60 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sS 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:42 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.013s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
443/tcp  open  https
1900/tcp open  upnp
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 1.62 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sT 192.168.0.1

Starting Nmap 7.60 ( https://nmap.org ) at 2023-09-01 11:43 IST
Nmap scan report for _gateway (192.168.0.1)
Host is up (0.011s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
443/tcp  open  https
1900/tcp open  upnp
MAC Address: AC:15:A2:B9:9E:29 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.39 seconds
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo nmap -sT 192.168.0.1
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
12:07:19.516119 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 3874417060 ecr 4037891199], length 87: HTTP: GE
T / HTTP/1.1
12:07:19.755206 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 0
12:07:19.755338 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [P.], seq 1:149, ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 148: HTTP:
 HTTP/1.1 204 No Content
12:07:19.755374 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.755589 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [F.], seq 88, ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.756033 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 4037891483 ecr 3874417060], length 0
12:07:19.756082 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 150, win 501, options [nop,nop,TS val 3874417300 ecr 4037891483], length 0
12:07:19.994677 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 89, win 506, options [nop,nop,TS val 4037891721 ecr 3874417299], length 0
12:12:19.267710 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [S], seq 3869803189, win 64240, options [mss 1460,sackOK,TS val 4275911588 ecr 0,nop,wscale 7], length
 0
12:12:19.392456 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [S.], seq 1016117305, ack 3869803190, win 65160, options [mss 1440,sackOK,TS val 1294573675 ecr 427591
1588,nop,wscale 7], length 0
12:12:19.392527 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 4275911712 ecr 1294573675], length 0
12:12:19.392729 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 4275911713 ecr 1294573675], length 87: HTTP: G
ET / HTTP/1.1
12:12:19.517262 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [P.], seq 1:190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 189: HTTP
: HTTP/1.1 204 No Content
12:12:19.517320 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 190, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.517396 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [F.], seq 190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 0
12:12:19.517543 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [F.], seq 88, ack 191, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.641753 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [.], ack 89, win 509, options [nop,nop,TS val 1294573924 ecr 4275911837], length 0
^C
22 packets captured
22 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 80
[sudo] password for lab1006:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:22:51.239093 IP 192.168.0.181.59480 > 192.168.0.1.80: Flags [.], ack 2427833317, win 1024, length 0
12:22:51.240122 IP 192.168.0.1.80 > 192.168.0.181.59480: Flags [R], seq 2427833317, win 0, length 0
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n 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
^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$
```

```
Activities  Terminal  Fri 12:31
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n 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
12:07:09.081766 IP 192.168.0.181.45849 > 192.168.0.1.80: Flags [.], ack 2996676280, win 1024, length 0
12:07:09.082405 IP 192.168.0.1.80 > 192.168.0.181.45849: Flags [R], seq 2996676280, win 0, length 0
12:07:19.275767 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [S], seq 1279895593, win 64240, options [mss 1460,sackOK,TS val 3874416820 ecr 0,nop,wscale 7], length
0
12:07:19.515835 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [S.], seq 1946168769, ack 1279895594, win 64768, options [mss 1420,sackOK,TS val 4037891199 ecr 3874416
820,nop,wscale 7], length 0
12:07:19.515903 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 3874417060 ecr 4037891199], length 0
12:07:19.516119 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 3874417060 ecr 4037891199], length 87: HTTP: GE
T / HTTP/1.1
12:07:19.755206 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 0
12:07:19.755338 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [P.], seq 1:149, ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 148: HTTP:
 HTTP/1.1 204 No Content
12:07:19.755374 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.755589 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [F.], seq 88, ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.756033 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 4037891483 ecr 3874417060], length 0
12:07:19.756082 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 150, win 501, options [nop,nop,TS val 3874417300 ecr 4037891483], length 0
12:07:19.994677 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 89, win 506, options [nop,nop,TS val 4037891721 ecr 3874417299], length 0
12:12:19.267710 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [S], seq 3869803189, win 64240, options [mss 1460,sackOK,TS val 4275911588 ecr 0,nop,wscale 7], length
 0
12:12:19.392456 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [S.], seq 1016117305, ack 3869803190, win 65160, options [mss 1440,sackOK,TS val 1294573675 ecr 427591
1588,nop,wscale 7], length 0
12:12:19.392527 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 4275911712 ecr 1294573675], length 0
12:12:19.392729 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 4275911713 ecr 1294573675], length 87: HTTP: G
ET / HTTP/1.1
12:12:19.517262 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [P.], seq 1:190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 189: HTTP
: HTTP/1.1 204 No Content
12:12:19.517320 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 190, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.517396 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [F.], seq 190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 0
12:12:19.517543 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [F.], seq 88, ack 191, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.641753 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [.], ack 89, win 509, options [nop,nop,TS val 1294573924 ecr 4275911837], length 0
^C
22 packets captured
22 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 80
[sudo] password for lab1006:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:22:51.239093 IP 192.168.0.181.59480 > 192.168.0.1.80: Flags [.], ack 2427833317, win 1024, length 0
12:22:51.240122 IP 192.168.0.1.80 > 192.168.0.181.59480: Flags [R], seq 2427833317, win 0, length 0
^C
```

```
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 445
[sudo] password for lab1006:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:06:51.203178 IP 192.168.0.181.61140 > 192.168.0.1.445: Flags [.], ack 764217825, win 1024, length 0
12:06:51.204633 IP 192.168.0.1.445 > 192.168.0.181.61140: Flags [R], seq 764217825, win 0, length 0
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n 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
12:07:09.081766 IP 192.168.0.181.45849 > 192.168.0.1.80: Flags [.], ack 2996676280, win 1024, length 0
12:07:09.082405 IP 192.168.0.1.80 > 192.168.0.181.45849: Flags [R], seq 2996676280, win 0, length 0
12:07:19.275767 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [S], seq 1279895593, win 64240, options [mss 1460,sackOK,TS val 3874416820 ecr 0,nop,wscale 7], length
0
12:07:19.515835 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [S.], seq 1946168769, ack 1279895594, win 64768, options [mss 1420,sackOK,TS val 4037891199 ecr 3874416
820,nop,wscale 7], length 0
12:07:19.515903 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 3874417060 ecr 4037891199], length 0
12:07:19.516119 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 3874417060 ecr 4037891199], length 87: HTTP: GE
T / HTTP/1.1
12:07:19.755206 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 0
12:07:19.755338 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [P.], seq 1:149, ack 88, win 506, options [nop,nop,TS val 4037891482 ecr 3874417060], length 148: HTTP:
 HTTP/1.1 204 No Content
12:07:19.755374 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.755589 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [F.], seq 88, ack 149, win 501, options [nop,nop,TS val 3874417299 ecr 4037891482], length 0
12:07:19.756033 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 4037891483 ecr 3874417060], length 0
12:07:19.756082 IP 192.168.0.181.34940 > 35.224.170.84.80: Flags [.], ack 150, win 501, options [nop,nop,TS val 3874417300 ecr 4037891483], length 0
12:07:19.994677 IP 35.224.170.84.80 > 192.168.0.181.34940: Flags [.], ack 89, win 506, options [nop,nop,TS val 4037891721 ecr 3874417299], length 0
12:12:19.267710 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [S], seq 3869803189, win 64240, options [mss 1460,sackOK,TS val 4275911588 ecr 0,nop,wscale 7], length
 0
12:12:19.392456 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [S.], seq 1016117305, ack 3869803190, win 65160, options [mss 1440,sackOK,TS val 1294573675 ecr 427591
1588,nop,wscale 7], length 0
12:12:19.392527 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 4275911712 ecr 1294573675], length 0
12:12:19.392729 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 4275911713 ecr 1294573675], length 87: HTTP: G
ET / HTTP/1.1
12:12:19.517262 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [P.], seq 1:190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 189: HTTP
: HTTP/1.1 204 No Content
12:12:19.517320 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [.], ack 190, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.517396 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [F.], seq 190, ack 88, win 509, options [nop,nop,TS val 1294573799 ecr 4275911713], length 0
12:12:19.517543 IP 192.168.0.181.37266 > 185.125.190.18.80: Flags [F.], seq 88, ack 191, win 501, options [nop,nop,TS val 4275911837 ecr 1294573799], length 0
12:12:19.641753 IP 185.125.190.18.80 > 192.168.0.181.37266: Flags [.], ack 89, win 509, options [nop,nop,TS val 1294573924 ecr 4275911837], length 0
^C
```

```
^C
18 packets captured
18 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 445
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:48:04.627527 IP 192.168.0.181.34783 > 192.168.0.1.445: Flags [none], win 1024, length 0
11:48:04.628128 IP 192.168.0.1.445 > 192.168.0.181.34783: Flags [R.], seq 0, ack 3263818715, win 0, length 0
11:52:06.632130 IP 192.168.0.181.60803 > 192.168.0.1.445: Flags [F], seq 957699247, win 1024, length 0
11:52:06.632639 IP 192.168.0.1.445 > 192.168.0.181.60803: Flags [R.], seq 0, ack 957699248, win 0, length 0
12:02:45.993860 IP 192.168.0.181.53342 > 192.168.0.1.445: Flags [FPU], seq 319605029, win 1024, urg 0, length 0
12:02:45.994487 IP 192.168.0.1.445 > 192.168.0.181.53342: Flags [R.], seq 0, ack 319605030, win 0, length 0
12:02:48.497744 IP 192.168.0.181.53353 > 192.168.0.1.445: Flags [FPU], seq 302827557, win 1024, urg 0, length 0
12:02:48.498182 IP 192.168.0.1.445 > 192.168.0.181.53353: Flags [R.], seq 0, ack 302827558, win 0, length 0
12:02:49.822022 IP 192.168.0.181.53354 > 192.168.0.1.445: Flags [FPU], seq 286051109, win 1024, urg 0, length 0
12:02:49.822529 IP 192.168.0.1.445 > 192.168.0.181.53354: Flags [R.], seq 0, ack 286051110, win 0, length 0
12:02:51.140952 IP 192.168.0.181.53355 > 192.168.0.1.445: Flags [FPU], seq 269273637, win 1024, urg 0, length 0
12:02:51.141450 IP 192.168.0.1.445 > 192.168.0.181.53355: Flags [R.], seq 0, ack 269273638, win 0, length 0
12:02:52.392377 IP 192.168.0.181.53356 > 192.168.0.1.445: Flags [FPU], seq 386714917, win 1024, urg 0, length 0
12:02:52.392785 IP 192.168.0.1.445 > 192.168.0.181.53356: Flags [R.], seq 0, ack 386714918, win 0, length 0
12:02:53.671027 IP 192.168.0.181.53357 > 192.168.0.1.445: Flags [FPU], seq 369937445, win 1024, urg 0, length 0
12:02:53.671644 IP 192.168.0.1.445 > 192.168.0.181.53357: Flags [R.], seq 0, ack 369937446, win 0, length 0
12:02:54.922803 IP 192.168.0.181.53358 > 192.168.0.1.445: Flags [FPU], seq 353160997, win 1024, urg 0, length 0
12:02:54.923238 IP 192.168.0.1.445 > 192.168.0.181.53358: Flags [R.], seq 0, ack 353160998, win 0, length 0
12:06:35.157721 IP 192.168.0.181.43512 > 192.168.0.1.445: Flags [.], ack 3238169869, win 1024, length 0
12:06:35.158256 IP 192.168.0.1.445 > 192.168.0.181.43512: Flags [R], seq 3238169869, win 0, length 0
^C
20 packets captured
20 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 445
[sudo] password for lab1006:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
12:06:51.203178 IP 192.168.0.181.61140 > 192.168.0.1.445: Flags [.], ack 764217825, win 1024, length 0
12:06:51.204633 IP 192.168.0.1.445 > 192.168.0.181.61140: Flags [R], seq 764217825, win 0, length 0
^C
2 packets captured
2 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n 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
12:07:09.081766 IP 192.168.0.181.45849 > 192.168.0.1.80: Flags [.], ack 2996676280, win 1024, length 0
```

```
Activities  Terminal  Fri 12:30
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
4 packets captured
4 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n 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
11:43:47.058689 IP 192.168.0.181.37582 > 192.168.0.1.80: Flags [S], seq 4035511020, win 64240, options [mss 1460,sackOK,TS val 734082481 ecr 0,nop,wscale 7], length 0
11:43:47.059295 IP 192.168.0.1.80 > 192.168.0.181.37582: Flags [S.], seq 2112648240, ack 4035511021, win 14480, options [mss 1460,sackOK,TS val 215401865 ecr 734082481,
nop,wscale 6], length 0
11:43:47.059348 IP 192.168.0.181.37582 > 192.168.0.1.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 734082481 ecr 215401865], length 0
11:43:47.059463 IP 192.168.0.181.37582 > 192.168.0.1.80: Flags [R.], seq 1, ack 1, win 502, options [nop,nop,TS val 734082481 ecr 215401865], length 0
11:44:19.052400 IP 192.168.0.181.50540 > 192.168.0.1.80: Flags [none], win 1024, length 0
11:44:19.152703 IP 192.168.0.181.50541 > 192.168.0.1.80: Flags [none], win 1024, length 0
11:47:19.338533 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [S], seq 3684885045, win 64240, options [mss 1460,sackOK,TS val 1543198147 ecr 0,nop,wscale 7], length
0
11:47:20.344923 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [S], seq 3684885045, win 64240, options [mss 1460,sackOK,TS val 1543199154 ecr 0,nop,wscale 7], length
0
11:47:20.582782 IP 34.122.121.32.80 > 192.168.0.181.52388: Flags [S.], seq 1538818171, ack 3684885046, win 64768, options [mss 1420,sackOK,TS val 2106952462 ecr 1543199
154,nop,wscale 7], length 0
11:47:20.582851 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 1543199392 ecr 2106952462], length 0
11:47:20.583053 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 1543199392 ecr 2106952462], length 87: HTTP: GE
T / HTTP/1.1
11:47:20.821843 IP 34.122.121.32.80 > 192.168.0.181.52388: Flags [.], ack 88, win 506, options [nop,nop,TS val 2106952748 ecr 1543199392], length 0
11:47:20.927208 IP 34.122.121.32.80 > 192.168.0.181.52388: Flags [P.], seq 1:149, ack 88, win 506, options [nop,nop,TS val 2106952855 ecr 1543199392], length 148: HTTP:
 HTTP/1.1 204 No Content
11:47:20.927261 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [.], ack 149, win 501, options [nop,nop,TS val 1543199736 ecr 2106952855], length 0
11:47:20.927474 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [F.], seq 88, ack 149, win 501, options [nop,nop,TS val 1543199736 ecr 2106952855], length 0
11:47:20.928590 IP 34.122.121.32.80 > 192.168.0.181.52388: Flags [F.], seq 149, ack 88, win 506, options [nop,nop,TS val 2106952856 ecr 1543199392], length 0
11:47:20.928639 IP 192.168.0.181.52388 > 34.122.121.32.80: Flags [.], ack 150, win 501, options [nop,nop,TS val 1543199737 ecr 2106952856], length 0
11:47:21.165183 IP 34.122.121.32.80 > 192.168.0.181.52388: Flags [.], ack 89, win 506, options [nop,nop,TS val 2106953093 ecr 1543199736], length 0
^C
18 packets captured
18 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 445
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:48:04.627527 IP 192.168.0.181.34783 > 192.168.0.1.445: Flags [none], win 1024, length 0
11:48:04.628128 IP 192.168.0.1.445 > 192.168.0.181.34783: Flags [R.], seq 0, ack 3263818715, win 0, length 0
11:52:06.632130 IP 192.168.0.181.60803 > 192.168.0.1.445: Flags [F], seq 957699247, win 1024, length 0
11:52:06.632639 IP 192.168.0.1.445 > 192.168.0.181.60803: Flags [R.], seq 0, ack 957699248, win 0, length 0
12:02:45.993860 IP 192.168.0.181.53342 > 192.168.0.1.445: Flags [FPU], seq 319605029, win 1024, urg 0, length 0
12:02:45.994487 IP 192.168.0.1.445 > 192.168.0.181.53342: Flags [R.], seq 0, ack 319605030, win 0, length 0
12:02:48.497744 IP 192.168.0.181.53353 > 192.168.0.1.445: Flags [FPU], seq 302827557, win 1024, urg 0, length 0
12:02:48.498182 IP 192.168.0.1.445 > 192.168.0.181.53353: Flags [R.], seq 0, ack 302827558, win 0, length 0
```

```
Activities  Terminal  Fri 12:30
lab1006@lab1006-HP-280-G4-MT-Business-PC: ~
File  Edit  View  Search  Terminal  Help
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 80
[sudo] password for lab1006:
Sorry, try again.
[sudo] password for lab1006:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:42:16.120897 IP 192.168.0.181.51304 > 142.250.192.35.80: Flags [.], ack 2968087265, win 501, options [nop,nop,TS val 1311487865 ecr 616512187], length 0
11:42:16.123483 IP 142.250.192.35.80 > 192.168.0.181.51304: Flags [.], ack 1, win 265, options [nop,nop,TS val 616522427 ecr 1311406277], length 0
11:42:19.274293 IP 192.168.0.181.34150 > 185.125.190.48.80: Flags [S], seq 694070017, win 64240, options [mss 1460,sackOK,TS val 1870985943 ecr 0,nop,wscale 7], length
0
11:42:19.403293 IP 185.125.190.48.80 > 192.168.0.181.34150: Flags [S.], seq 57117681, ack 694070018, win 65160, options [mss 1440,sackOK,TS val 2095797192 ecr 187098594
3,nop,wscale 7], length 0
11:42:19.403362 IP 192.168.0.181.34150 > 185.125.190.48.80: Flags [.], ack 1, win 502, options [nop,nop,TS val 1870986072 ecr 2095797192], length 0
11:42:19.403550 IP 192.168.0.181.34150 > 185.125.190.48.80: Flags [P.], seq 1:88, ack 1, win 502, options [nop,nop,TS val 1870986072 ecr 2095797192], length 87: HTTP: G
ET / HTTP/1.1
11:42:19.532894 IP 185.125.190.48.80 > 192.168.0.181.34150: Flags [P.], seq 1:190, ack 88, win 509, options [nop,nop,TS val 2095797321 ecr 1870986072], length 189: HTTP
: HTTP/1.1 204 No Content
11:42:19.532952 IP 192.168.0.181.34150 > 185.125.190.48.80: Flags [.], ack 190, win 501, options [nop,nop,TS val 1870986202 ecr 2095797321], length 0
11:42:19.532973 IP 185.125.190.48.80 > 192.168.0.181.34150: Flags [F.], seq 190, ack 88, win 509, options [nop,nop,TS val 2095797321 ecr 1870986072], length 0
11:42:19.533163 IP 192.168.0.181.34150 > 185.125.190.48.80: Flags [F.], seq 88, ack 191, win 501, options [nop,nop,TS val 1870986202 ecr 2095797321], length 0
11:42:19.661772 IP 185.125.190.48.80 > 192.168.0.181.34150: Flags [.], ack 89, win 509, options [nop,nop,TS val 2095797450 ecr 1870986202], length 0
11:42:26.360898 IP 192.168.0.181.51304 > 142.250.192.35.80: Flags [.], ack 1, win 501, options [nop,nop,TS val 1311498105 ecr 616522427], length 0
11:42:26.363148 IP 142.250.192.35.80 > 192.168.0.181.51304: Flags [.], ack 1, win 265, options [nop,nop,TS val 616532667 ecr 1311406277], length 0
11:42:36.601010 IP 192.168.0.181.51304 > 142.250.192.35.80: Flags [.], ack 1, win 501, options [nop,nop,TS val 1311508345 ecr 616532667], length 0
11:42:36.604019 IP 142.250.192.35.80 > 192.168.0.181.51304: Flags [.], ack 1, win 265, options [nop,nop,TS val 616542908 ecr 1311406277], length 0
11:42:39.275106 IP 192.168.0.181.40282 > 192.168.0.1.80: Flags [S], seq 3619155204, win 1024, options [mss 1460], length 0
11:42:39.276007 IP 192.168.0.1.80 > 192.168.0.181.40282: Flags [S.], seq 3810892068, ack 3619155205, win 14600, options [mss 1460], length 0
11:42:39.276063 IP 192.168.0.181.40282 > 192.168.0.1.80: Flags [R], seq 3619155205, win 0, length 0
^C
18 packets captured
18 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 81
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp3s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:42:57.611975 IP 192.168.0.181.47830 > 192.168.0.1.81: Flags [S], seq 2104214126, win 1024, options [mss 1460], length 0
11:42:57.613623 IP 192.168.0.1.81 > 192.168.0.181.47830: Flags [R.], seq 0, ack 2104214127, win 0, length 0
11:43:08.868650 IP 192.168.0.181.40980 > 192.168.0.1.81: Flags [S], seq 3031532005, win 64240, options [mss 1460,sackOK,TS val 734044290 ecr 0,nop,wscale 7], length 0
11:43:08.872715 IP 192.168.0.1.81 > 192.168.0.181.40980: Flags [R.], seq 0, ack 3031532006, win 0, length 0
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
lab1006@lab1006-HP-280-G4-MT-Business-PC:~$ sudo tcpdump -n port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
```

Conclusion:-

Nmap offers a diverse set of scanning techniques to suit various network reconnaissance needs. The choice of scan depends on factors like stealth, speed, and the specific information you seek. Understanding these techniques is vital for network administrators and security professionals to safeguard their networks and systems from potential threats

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 06/09/2023

Lab Assignment No:-9

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 three-way 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

USE IP OF tsec.edu USING DIG

Output Screenshots:-

```
prasad@prasad-VirtualBox:~$ gedit sample.txt
prasad@prasad-VirtualBox:~$ sudo apt-get install hping3
[sudo] password for prasad:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  hping3
0 upgraded, 1 newly installed, 0 to remove and 48 not upgraded.
Need to get 107 kB of archives.
After this operation, 284 kB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu bionic/universe amd64 hping3 amd64 3.a2.ds2-7 [107 kB]
Fetched 107 kB in 1s (94.1 kB/s)
Selecting previously unselected package hping3.
(Reading database ... 165323 files and directories currently installed.)
Preparing to unpack .../hping3_3.a2.ds2-7_amd64.deb ...
Unpacking hping3 (3.a2.ds2-7) ...
Setting up hping3 (3.a2.ds2-7) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
prasad@prasad-VirtualBox:~$ man hping3
prasad@prasad-VirtualBox:~$ man hping3
prasad@prasad-VirtualBox:~$ hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159
[open_sockraw] socket(): Operation not permitted
[main] can't open raw socket
prasad@prasad-VirtualBox:~$ sudo su
[sudo] password for prasad:
root@prasad-VirtualBox:/home/prasad# ^C
root@prasad-VirtualBox:/home/prasad# hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159
HPING 192.168.1.159 (enp0s3 192.168.1.159): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.1.159 hping statistic ---
10859977 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@prasad-VirtualBox:/home/prasad# hping3 -1 --flood -a 192.168.103.1 192.168.1.255
HPING 192.168.1.255 (enp0s3 192.168.1.255): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.1.255 hping statistic ---
1175003 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
root@prasad-VirtualBox:/home/prasad#
```

```
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 525299007:525299127, win 64, length 120: HTTP
21:33:33.522520 IP 159.157.124.107.5556 > 192.168.1.159.80: Flags [S], seq 1481118578:1481118690, 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 117.237.227.248.5557 > 192.168.1.159.80: Flags [S], seq 125305799:125305919, 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 169809196:169809316, win 64, length 120: HTTP
21:33:33.589008 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
21:33:33.626066 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 37.189.125.148.5572 > 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 54.110.229.124.5575 > 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 1292070229:1292207149, win 64, length 120: HTTP
21:33:33.723087 IP 248.217.122.89.5577 > 192.168.1.159.80: Flags [S], seq 1695026106:1695026226, 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
21:33:33.744034 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
21:33:33.778323 IP 47.71.231.2.5582 > 192.168.1.159.80: Flags [S], seq 127606895:127607015, win 64, length 120: HTTP
21:33:33.778349 IP 64.105.171.116.5586 > 192.168.1.159.80: Flags [S], seq 393678716:393678836, 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
21:33:33.702417 IP 0.159.158.54.5590 > 192.168.1.159.80: Flags [S], seq 363642928:363643048, win 64, length 120: HTTP
21:33:33.704203 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 0.0.0.0.68 > 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 315108362:315108422, 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 171.106.185.119.46025 > 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 568158981:568159021, 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 1143484615:1143484735, 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 231.200.95.228.46042 > 192.168.1.159.80: Flags [S], seq 1883924015:1883924135, 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
21:33:54.910739 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 106.93.58.99.46247 > 192.168.1.159.80: Flags [S], seq 72931271:72931391, win 64, length 120: HTTP
21:33:54.943996 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
21:33:54.969485 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 114.124.68.109.46352 > 192.168.1.159.80: Flags [S], seq 1035223507:1035223627, win 64, length 120: HTTP
21:33:55.006566 IP 65.13.111.224.46398 > 192.168.1.159.80: Flags [S], seq 76909556:76989676, win 64, length 120: HTTP
21:33:55.017702 IP 195.149.140.48.46255 > 192.168.1.159.80: Flags [S], seq 166081009:166081129, win 64, length 120: HTTP
21:33:55.037174 IP 225.92.6.112.46258 > 192.168.1.159.80: Flags [S], seq 1493974638:1493974750, win 64, length 120: HTTP
21:33:55.048803 IP 64.151.104.180.46259 > 192.168.1.159.80: Flags [S], seq 1352651598:1352651718, win 64, length 120: HTTP
21:33:55.053192 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 1772187208:1772187320, 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
21:33:55.097900 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
21:33:55.232640 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 24782, 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
21:35:28.465588 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
21:35:28.495669 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

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 13/09/2023

Lab Assignment No:-10

Aim:- To study and configure firewalls using IP tables

Lab Outcome Attained :- LO6

Theory:-

1. What is firewall ?

A firewall is a security system designed to prevent unauthorized access to or from a private network. It can be implemented as hardware, software, or a combination of both. Typically, a firewall has two network interfaces: one facing the external network and one facing the internal network. Its primary function is to regulate and control the traffic that can pass from one side to the other.

At its most fundamental level, firewalls can block network traffic intended for specific IP addresses or server ports. In a TCP network, data is transmitted in packets, each comprising a packet header that contains essential control information, including source and destination addresses and packet sequence details, along with the actual data payload. This control information within each packet not only ensures the proper delivery of associated data but also provides firewalls with various criteria for matching packets against predefined firewall rules.

2. List different types of firewall .

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.

3.Write different options that can be used in configuring firewall.

Configuring a firewall using IPtables involves a series of operations to define rules and policies that control network traffic. IPtables is a popular firewall management tool on Linux systems.Different options involved in configuring a firewall using IPtables include:

Listing Existing Rules:

You can start by listing existing rules to understand the current configuration. Use the iptables -L or iptables -S command to view the existing rules in place.

Setting Default Policies:

Define the default policies for incoming and outgoing traffic. This sets the default action (ACCEPT, DROP, or REJECT) for packets that don't match any specific rules.

Creating Rules:

You can create rules to allow or block traffic based on various criteria such as source IP, destination IP, ports, and protocols. For example, you can use the iptables -A command to append a rule.

Specifying Chains:
IPtables uses chains (e.g., INPUT, OUTPUT, FORWARD) to organize rules. You can specify which chain a rule applies to when creating it.

Defining Match Criteria:
Define match criteria for each rule. You can use parameters like -s for source IP, -d for destination IP, -p for protocol, and -m for matching modules.

Defining Target Actions:
Specify the target action for matched packets. Common targets include ACCEPT, DROP (silently discard), REJECT (discard with an error message), and LOG (log the packet).

Modifying Rules:
You can modify existing rules using commands like iptables -R (replace), iptables -D (delete), or iptables -I (insert) to make changes to specific rules.

Saving and Restoring Rules:
After configuring rules, use the iptables-save command to save the rules to a file. You can later restore the rules using the iptables-restore command.

Network Address Translation (NAT):
If you need to set up port forwarding, masquerading, or other NAT-related tasks, you can configure NAT rules using IPtables.

Packet Filtering for Specific Applications:
You can configure rules that are specific to certain applications or services, such as web servers (HTTP and HTTPS), DNS, or SSH.

4.Write the commands used for configuring firewall using IPTABLES.

a) List Existing Rules:

iptables -L or iptables --list: List current rules.
iptables -S or iptables --list-rules: Show rules in a more detailed format.

Set Default Policies:

iptables -P [chain] [target]: Set the default policy for a chain (e.g., INPUT, OUTPUT, FORWARD).

Create Rules:

iptables -A [chain] [options]: Append a rule to a chain.
iptables -I [chain] [rule number] [options]: Insert a rule at a specific position.
iptables -N [chain]: Create a new user-defined chain.

Specify Chains:

Use the -A, -I, or -N options with the desired chain name.

Define Match Criteria:

-s [source]: Match packets from a specific source.
-d [destination]: Match packets going to a specific destination.
-p [protocol]: Match packets of a specific protocol.
-m [module] [options]: Use matching modules for more complex criteria.
Define Target Actions:
-j [target]: Specify the target action (e.g., ACCEPT, DROP, REJECT, LOG) for matched packets.

Modify Rules:

iptables -R [chain] [rule number] [options]: Replace a rule.
iptables -D [chain] [options]: Delete a rule.
iptables -I [chain] [rule number] [options]: Insert a rule.

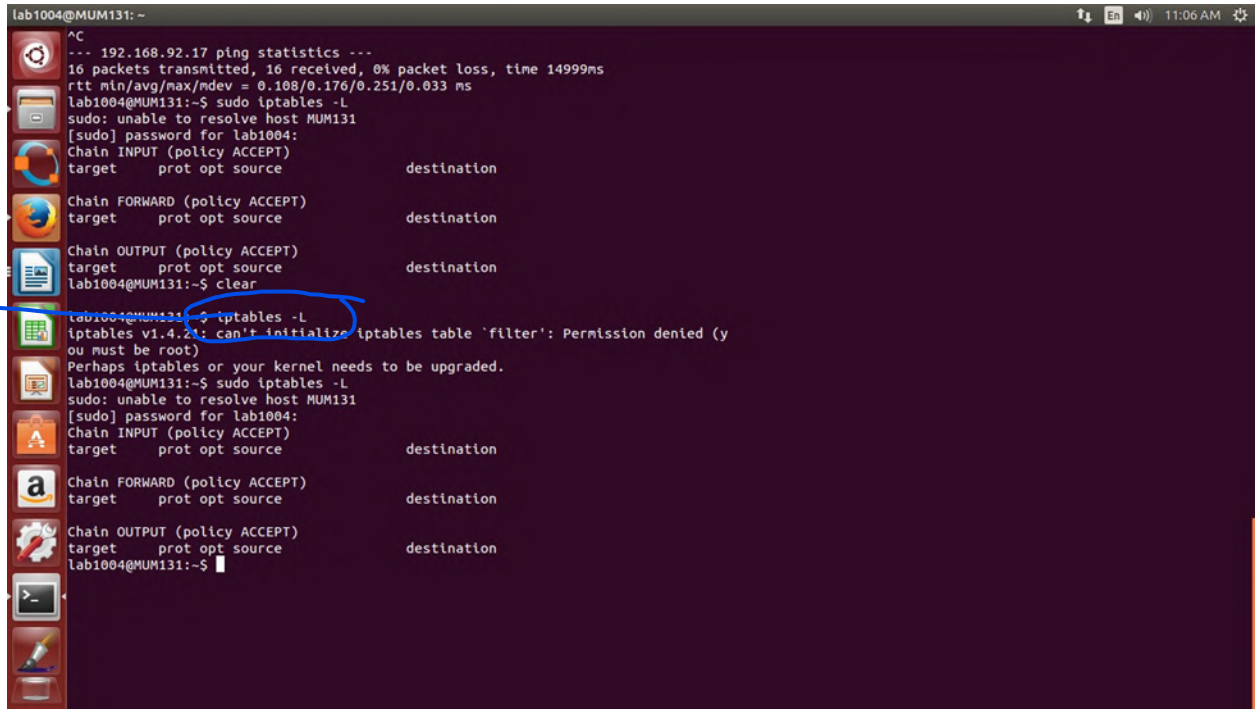
Save and Restore Rules:

iptables-save: Save rules to a file (typically redirected to a file).

iptables-restore: Restore rules from a file.

Output Screenshots:-

START



```
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:~$
```

```
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:~$
```

```
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:~$
```

Conclusion:-

Learnt about firewall , types of firewall , configuration of firewall using rules and also executed basic options for firewall like allowing incoming traffic on specific port , blocking traffic , editing IP tables.

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 06/09/2023

~~Lab Assignment No:-11~~

Aim:- Installing snort, setting in Intrusion Detection Mode and writing rules for Intrusion Detection

Lab Outcome Attained :- LO6

Theory:-

What is Intrusion Detection System?

1. A system called an intrusion detection system (IDS) observes network traffic for malicious transactions and sends immediate alerts when it is observed. It is software
that checks a network or system for malicious activities or policy violations.
2. Each illegal activity or violation is often recorded either centrally using a SIEM system
or notified to an administration.
3. IDS monitors a network or system for malicious activity and protects a computer network from unauthorized access from users, including perhaps insiders.
4. The intrusion detector learning task is to build a predictive model (i.e. a classifier)
capable of distinguishing between 'bad connections' (intrusion/attacks) and 'good (normal) connections'.

What are different modes in which Snort works?

Snort operates in three primary modes:

1. Sniffer Mode:

In this mode, Snort acts as a packet sniffer, analyzing network traffic and displaying
the captured packets on the console. It doesn't perform any active intrusion detection or prevention; instead, it's used for network analysis and troubleshooting purposes.

2. Packet Logger Mode:
In packet logger mode, Snort captures and logs network traffic that matches defined rules to log files. This mode is useful for creating a record of network activity for later analysis.

3. Network Intrusion Detection System (NIDS) Mode:
This is the main mode of Snort, where it functions as a network intrusion detection system (NIDS). Snort examines network traffic against a set of predefined rules to identify and alert on potential intrusion attempts, malicious activities, or suspicious patterns. When a rule matches, Snort generates alerts that can be sent to various destinations, such as log files, syslog servers, or email.
In addition to these primary modes, Snort also offers inline capabilities through its IPS
(Intrusion Prevention System) mode:

4. Network Intrusion Prevention System (IPS) Mode:
When operating as an IPS, Snort not only detects suspicious activity but can also take active measures to prevent or block potential threats. In this mode, Snort can drop or modify packets that match specific rules, effectively preventing malicious traffic from reaching its intended target.

Write the commands used for installing snort, editing its configuration file and configuring it in Intrusion Detection Mode?

1.Check the name of the interface using command ifconfig.
Installing Snort

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
Editing Configuration File (snort.conf):

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 192.168.44.0/24 (in
section 1)

6. Open new terminal. Open ftp.rulefile 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 ens33 to validate that all rules are there.

8. Type the command sudo snort -A console -q -u snort -g snort -c /etc/snort/snort.conf
-iens33

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

10. 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;)

11. Add the local.rules file in section7 of configuration file of snort by writing: include
$RULE_PATH local.rules

12. Validate the changes made in snort.conf file by writing the command in terminal: sudo
snort -T -c /etc/snort/snort.conf -iens33

13. 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 -iens33

Conclusion:-

Snort as an Intrusion Detection System(IDS) play a important part in enhancing network security . Snort works in several modes which include sniffer , packet logger , NIDS mode .We installed snort step by step and configured snort for our own system to make it compatible with our system and also we can do full configuration for various security needs

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 13/09/2023

Lab Assignment No:-~~13~~ 12

Aim:- Explore the GPG tool of Linux to implement email security

Lab Outcome Attained :- LO6

Theory:-

What is private key ring and public key ring ?

a)Public key ring

The public key ring contains the public keys of other users. These keys are made available to the public so that anyone can encrypt messages to the user. The public key ring is typically shared with other users by exporting it to a file or by adding it to a PGP keyserver.

The public key contains the following information:

The user's name or email address
The user's fingerprint, which is a unique identifier for the key
The key's algorithm and strength
The key's expiration date
When someone wants to encrypt a message to you, they will use your public key. The message will be encrypted using the public key, but it can only be decrypted using the corresponding private key.

b)Private key ring

The private key ring contains the private keys of the user. These keys are kept secret and should not be shared with anyone. The private key ring is typically protected by a password or passphrase.

The private key contains the following information:

The user's name or email address
The user's fingerprint, which is a unique identifier for the key
The key's algorithm and strength
The key's expiration date
The private key is used to decrypt messages that have been encrypted with the user's public key. It is also used to sign messages, which allows the recipient to verify that the message was sent by the intended sender.

The public key ring and the private key ring are essential for using PGP. They allow users to encrypt and decrypt messages securely.

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

Key generation

The following command generates a new GPG key pair:

START

gpg --gen-key
This command will prompt you for some information, such as your name, email address, and key length.

Export and import of keys

The following command exports the public key to a file:

gpg --export --output public.key
The following command imports the public key from a file:

gpg --import public.key
The following command exports the private key to a file:

gpg --export-secret-key --output private.key
The following command imports the private key from a file:

gpg --import-secret-key private.key
Signing and encrypting the message

The following command signs a message:

gpg --sign message.txt
The following command encrypts a message:

gpg --encrypt --recipient recipient@example.com message.txt
The recipient can then decrypt the message using their private key.

Some additional details about the commands:

The gpg command is the main GPG command.
The **--gen-key** option generates a new GPG key pair.
The **--export** option exports a key to a file.
The **--import** option imports a key from a file.
The **--sign** option signs a message.
The **--encrypt** option encrypts a message.
The **--recipient** option specifies the recipient of the encrypted message.

Output Screenshots

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help
12:12:12.507575 IP 192.168.1.103 > 192.168.1.225: ICMP echo request,
, seq 3927, length 8
12:12:12.507577 IP 192.168.1.103 > 192.168.1.225: ICMP echo request,
, seq 4183, length 8
12:12:12.507660 IP 192.168.1.103 > 192.168.1.225: ICMP echo request,
, seq 4439, length 8
12:12:12.507679 IP 192.168.1.103 > 192.168.1.225: ICMP echo request,
, seq 4695, length 8
12:12:12.548432 ARP, Request who-has 192.168.69.2 tell 192.168.69.128
 28
12:12:12.786567 ARP, Request who-has 192.168.1.103 tell 192.168.69.2,
46
12:12:13.573489 ARP, Request who-has 192.168.69.2 tell 192.168.69.128
 28
12:12:13.791642 ARP, Request who-has 192.168.1.103 tell 192.168.69.2,
46
12:12:14.597360 ARP, Request who-has 192.168.69.2 tell 192.168.69.128
 28
12:12:14.786963 ARP, Request who-has 192.168.1.103 tell 192.168.69.2,
46
^C
363813 packets captured
481095 packets received by filter
117282 packets dropped by kernel

┌──(root㉿kaliVirtual)-[~]
└─#
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help
hping3 version 3.0.0-alpha-2 ($Id: release.h,v 1.4 2004/04/09 23:38:56 antire
z Exp $)
This binary is TCL scripting capable

┌──(root㉿kaliVirtual)-[~]
└─# man hping3

┌──(root㉿kaliVirtual)-[~]
└─# hping3 -c 15000 -d 120 -S -w 64 -p 80 --rand-source 192.168.1.159
HPING 192.168.1.159 (eth0 192.168.1.159): S set, 40 headers + 120 data bytes
^C
--- 192.168.1.159 hping statistic ---
20 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

┌──(root㉿kaliVirtual)-[~]
└─# hping3 -1 --flood -a 192.168.1.103 192.168.1.225
HPING 192.168.1.225 (eth0 192.168.1.225): icmp mode set, 28 headers + 0 data
bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.1.225 hping statistic ---
481042 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

┌──(root㉿kaliVirtual)-[~]
└─#
```

root@kaliVirtual: ~

File Actions Edit View Help

```
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
gpg: WARNING: no command supplied.  Trying to guess what you mean ...
gpg: Go ahead and type your message ...
^C
gpg: signal Interrupt caught ... exiting


┌──(root㉿kaliVirtual)-[~]
└─# gpg --version
gpg (GnuPG) 2.2.40
libgcrypt 1.10.2
Copyright (C) 2022 g10 Code GmbH
License GNU GPL-3.0-or-later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /root/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2

┌──(root㉿kaliVirtual)-[~]
└─#
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

     -q, --quiet
             Try to be as quiet as possible.  Should not be used in a
             tion file.

     --batch
     --no-batch
             Use batch mode.  Never ask, do  not  allow  interactive
             mands.  --no-batch  disables this option.  Note that even
             a filename given on the command line, gpg might still ne
             read  from STDIN (in particular if gpg figures that the
             is a detached signature and no data file has been specif
             Thus if you do not want to feed data via  STDIN,  you  s
             connect STDIN to '/dev/null'.

             It  is  highly  recommended to use this option along wit
             options --status-fd  and --with-colons  for any unattended
             of gpg.  Should not be used in an option file.

     --no-tty
             Make  sure that the TTY (terminal) is never used for any
             put.  This option is needed in some cases because GnuPG
             times prints warnings to the TTY even if --batch  is used

     --yes   Assume "yes" on most questions.  Should not be used in a
             tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

┌──(root㉿kaliVirtual)-[~]
└─# gpg --full-generate-key
gpg (GnuPG) 2.2.40; Copyright (C) 2022 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 1024
Requested keysize is 1024 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

     -q, --quiet
             Try to be as quiet as possible.  Should not be used in a
             tion file.

     --batch
     --no-batch
             Use batch mode.  Never ask, do  not  allow  interactive
             mands.  --no-batch  disables this option.  Note that even
             a filename given on the command line, gpg might still ne
             read  from STDIN (in particular if gpg figures that the
             is a detached signature and no data file has been specif
             Thus if you do not want to feed data via  STDIN,  you  s
             connect STDIN to '/dev/null'.

             It  is  highly  recommended to use this option along wit
             options --status-fd  and --with-colons  for any unattended
             of gpg.  Should not be used in an option file.

     --no-tty
             Make  sure that the TTY (terminal) is never used for any
             put.  This option is needed in some cases because GnuPG
             times prints warnings to the TTY even if --batch  is used

     --yes   Assume "yes" on most questions.  Should not be used in a
             tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

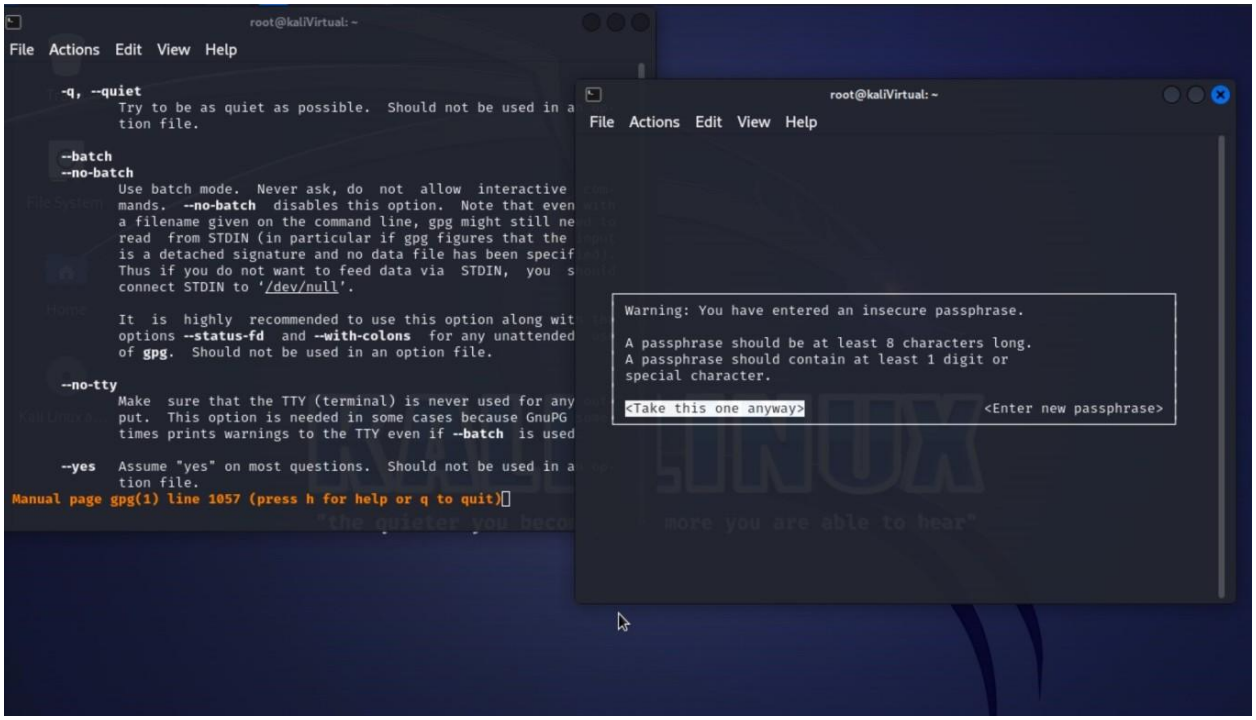
```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

Please enter the passphrase to
protect your new key

Passphrase:

      <OK>                    <Cancel>
```

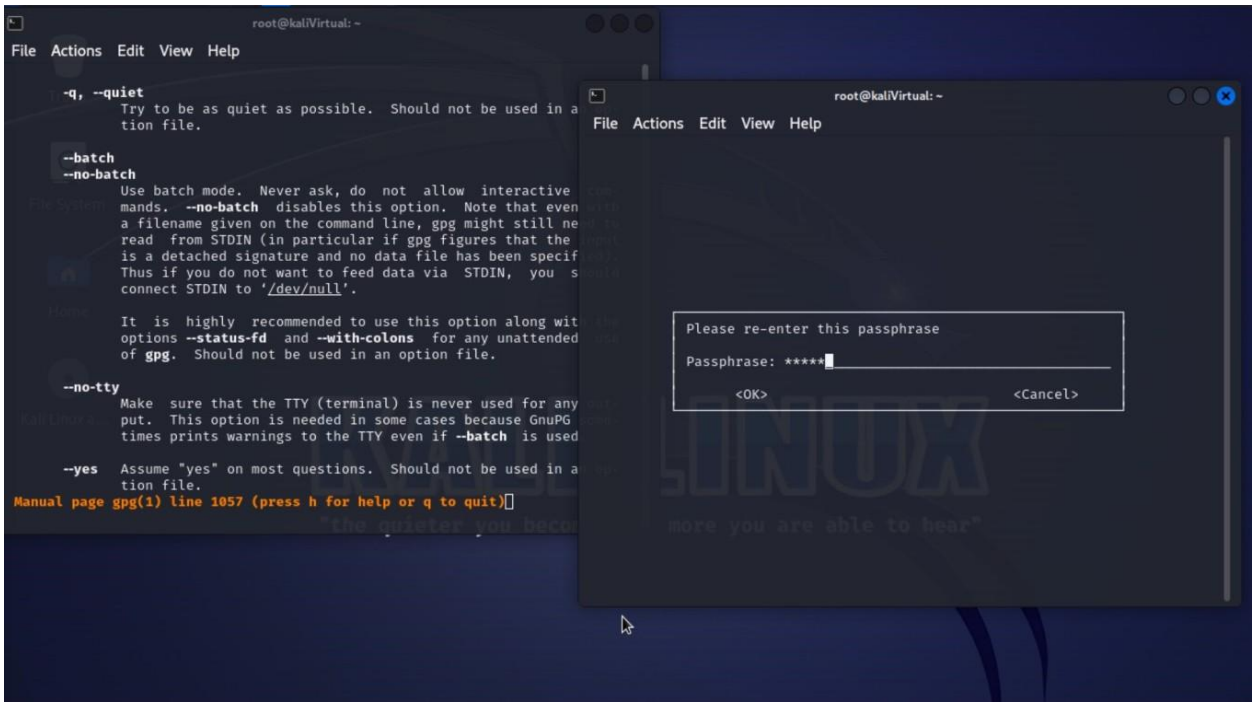
```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

      -q, --quiet
              Try to be as quiet as possible.  Should not be used in a
              tion file.

      --batch
      --no-batch
              Use batch mode.  Never ask, do  not  allow  interactive
              mands.  --no-batch  disables this option.  Note that even
              a filename given on the command line, gpg might still ne
              read  from STDIN (in particular if gpg figures that the
              is a detached signature and no data file has been specif
              Thus if you do not want to feed data via  STDIN,  you  s
              connect STDIN to '/dev/null'.

              It  is  highly  recommended to use this option along wit
              options --status-fd  and --with-colons  for any unattended
              of gpg.  Should not be used in an option file.

      --no-tty
              Make  sure that the TTY (terminal) is never used for any
              put.  This option is needed in some cases because GnuPG
              times prints warnings to the TTY even if --batch  is used

      --yes   Assume "yes" on most questions.  Should not be used in a
              tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help
      <n>y = key expires in n years
Key is valid for? (0) 2
Key expires at Fri Sep 15 10:53:12 2023 IST
Is this correct? (y/N) y

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

Real name: Pratham
Email address: pratham@abc.com
Comment: sender
You selected this USER-ID:
    "Pratham (sender) <pratham@abc.com>"

Change (N)ame, (C)omment, (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: /root/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/root/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/D4721C0C
22F006823B8C2A7DBBA44BFF508E371A.rev'
public and secret key created and signed.
```


```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

      -q, --quiet
              Try to be as quiet as possible.  Should not be used in a
              tion file.

      --batch
      --no-batch
              Use batch mode.  Never ask, do  not  allow  interactive
              mands.  --no-batch  disables this option.  Note that even
              a filename given on the command line, gpg might still ne
              read  from STDIN (in particular if gpg figures that the
              is a detached signature and no data file has been specif
              Thus if you do not want to feed data via  STDIN,  you  s
              connect STDIN to '/dev/null'.

              It  is  highly  recommended to use this option along wit
              options --status-fd  and --with-colons  for any unattended
              of gpg.  Should not be used in an option file.

      --no-tty
              Make  sure that the TTY (terminal) is never used for any
              put.  This option is needed in some cases because GnuPG
              times prints warnings to the TTY even if --batch  is used

      --yes   Assume "yes" on most questions.  Should not be used in a
              tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```


```
root@kaliVirtual: ~
File  Actions  Edit  View  Help
Comment: sender
You selected this USER-ID:
    "Pratham (sender) <pratham@abc.com>"

Change (N)ame, (C)omment, (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: /root/.gnupg/trustdb.gpg: trustdb created
gpg: directory '/root/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/D4721C0C
22F006823B8C2A7DBBA44BFF508E371A.rev'
public and secret key created and signed.

pub   rsa1024 2023-09-13 [SC] [expires: 2023-09-15]
      D4721C0C22F006823B8C2A7DBBA44BFF508E371A
uid                      Pratham (sender) <pratham@abc.com>
sub   rsa1024 2023-09-13 [E] [expires: 2023-09-15]

┌──(root㉿kaliVirtual)-[~]
└─#
```


root@kaliVirtual: ~
File Actions Edit View Help
-q, --quiet
Try to be as quiet as possible. Should not be used in a
tion file.
--batch
--no-batch
Use batch mode. Never ask, do not allow interactive
mands. --no-batch disables this option. Note that even
a filename given on the command line, gpg might still ne
read from STDIN (in particular if gpg figures that the
is a detached signature and no data file has been specif
Thus if you do not want to feed data via STDIN, you s
connect STDIN to '/dev/null'.
It is highly recommended to use this option along wit
options --status-fd and --with-colons for any unattended
of gpg. Should not be used in an option file.
--no-tty
Make sure that the TTY (terminal) is never used for any
put. This option is needed in some cases because GnuPG
times prints warnings to the TTY even if --batch is used
--yes Assume "yes" on most questions. Should not be used in a
tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
root@kaliVirtual: ~
File Actions Edit View Help
Warning: You have entered an insecure passphrase.
A passphrase should be at least 8 characters long.
A passphrase should contain at least 1 digit or
special character.
<Take this one anyway>
<Enter new passphrase>


root@kaliVirtual: ~
File Actions Edit View Help
-q, --quiet
Try to be as quiet as possible. Should not be used in a
tion file.
--batch
--no-batch
Use batch mode. Never ask, do not allow interactive
mands. --no-batch disables this option. Note that even
a filename given on the command line, gpg might still ne
read from STDIN (in particular if gpg figures that the
is a detached signature and no data file has been specif
Thus if you do not want to feed data via STDIN, you s
connect STDIN to '/dev/null'.
It is highly recommended to use this option along wit
options --status-fd and --with-colons for any unattended
of gpg. Should not be used in an option file.
--no-tty
Make sure that the TTY (terminal) is never used for any
put. This option is needed in some cases because GnuPG
times prints warnings to the TTY even if --batch is used
--yes Assume "yes" on most questions. Should not be used in a
tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
root@kaliVirtual: ~
File Actions Edit View Help
Please re-enter this passphrase
Passphrase: *****
<OK>
<Cancel>

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

     -q, --quiet
             Try to be as quiet as possible.  Should not be used in a
             tion file.

     --batch
     --no-batch
             Use batch mode.  Never ask, do  not  allow  interactive
             mands.  --no-batch  disables this option.  Note that even
             a filename given on the command line, gpg might still ne
             read  from STDIN (in particular if gpg figures that the
             is a detached signature and no data file has been specif
             Thus if you do not want to feed data via  STDIN,  you  s
             connect STDIN to '/dev/null'.

             It  is  highly  recommended to use this option along wit
             options --status-fd  and --with-colons  for any unattended
             of gpg.  Should not be used in an option file.

     --no-tty
             Make  sure that the TTY (terminal) is never used for any
             put.  This option is needed in some cases because GnuPG
             times prints warnings to the TTY even if --batch  is used

     --yes   Assume "yes" on most questions.  Should not be used in a
             tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help
└─# realpath prathampublic
/root/prathampublic

┌──(root㉿kaliVirtual)-[~]
└─# gpg --list-keys
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 2023-09-15
/root/.gnupg/pubring.kbx
-------------------------
pub   rsa1024 2023-09-13 [SC] [expires: 2023-09-15]
      D4721C0C22F006823B8C2A7DBBA44BFF508E371A
uid           [ultimate] Pratham (sender) <pratham@abc.com>
sub   rsa1024 2023-09-13 [E] [expires: 2023-09-15]

pub   rsa3072 2023-09-13 [SC] [expires: 2025-09-12]
      02F6CDDE0C3FA65F3481F436677A0AD657422C4C
uid           [ultimate] manav <manav@abc.com>
sub   rsa3072 2023-09-13 [E] [expires: 2025-09-12]


┌──(root㉿kaliVirtual)-[~]
└─# gpg --export -a manav>manavpublic

┌──(root㉿kaliVirtual)-[~]
└─#
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

     -q, --quiet
             Try to be as quiet as possible.  Should not be used in a
             tion file.

     --batch
     --no-batch
             Use batch mode.  Never ask, do  not  allow  interactive
             mands.  --no-batch  disables this option.  Note that even
             a filename given on the command line, gpg might still ne
             read  from STDIN (in particular if gpg figures that the
             is a detached signature and no data file has been specif
             Thus if you do not want to feed data via  STDIN,  you  s
             connect STDIN to '/dev/null'.

             It  is  highly  recommended to use this option along wit
             options --status-fd  and --with-colons  for any unattended
             of gpg.  Should not be used in an option file.

     --no-tty
             Make  sure that the TTY (terminal) is never used for any
             put.  This option is needed in some cases because GnuPG
             times prints warnings to the TTY even if --batch  is used

     --yes   Assume "yes" on most questions.  Should not be used in a
             tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

┌──(root㉿kaliVirtual)-[~]
└─# gpg --gen-key
gpg (GnuPG) 2.2.40; Copyright (C) 2022 g10 Code GmbH
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: manav
Email address: manav@abc.com
You selected this USER-ID:
    "manav <manav@abc.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.
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

     -q, --quiet
             Try to be as quiet as possible.  Should not be used in a
             tion file.

     --batch
     --no-batch
             Use batch mode.  Never ask, do  not  allow  interactive
             mands.  --no-batch  disables this option.  Note that even
             a filename given on the command line, gpg might still ne
             read  from STDIN (in particular if gpg figures that the
             is a detached signature and no data file has been specif
             Thus if you do not want to feed data via  STDIN,  you  s
             connect STDIN to '/dev/null'.

             It  is  highly  recommended to use this option along wit
             options --status-fd  and --with-colons  for any unattended
             of gpg.  Should not be used in an option file.

     --no-tty
             Make  sure that the TTY (terminal) is never used for any
             put.  This option is needed in some cases because GnuPG
             times prints warnings to the TTY even if --batch  is used

     --yes   Assume "yes" on most questions.  Should not be used in a
             tion file.
Manual page gpg(1) line 1057 (press h for help or q to quit)
```

```
root@kaliVirtual: ~
File  Actions  Edit  View  Help

Real name: manav
Email address: manav@abc.com
You selected this USER-ID:
    "manav <manav@abc.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: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/02F6CDDE
0C3FA65F3481F436677A0AD657422C4C.rev'
public and secret key created and signed.

pub   rsa3072 2023-09-13 [SC] [expires: 2025-09-12]
      02F6CDDE0C3FA65F3481F436677A0AD657422C4C
uid                      manav <manav@abc.com>
sub   rsa3072 2023-09-13 [E] [expires: 2025-09-12]

┌──(root㉿kaliVirtual)-[~]
└─#
```

Conclusion:-

Learnt about GPG tool in linux and how it provides email security , executed several commands related to GPG and also explored more about public key ring and private key rings

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 20/09/2023

Security Lab Theory Assignment 1

Lab Outcome Attained :- LO2

Explain the padding scheme used in RSA. Why it is used ? What is its limitation?

The padding scheme used in RSA, in given youtube video , is known as **PKCS (Public Key Cryptography Standards) version 1.5 padding.** This padding scheme is used to enhance the security of RSA encryption by introducing **randomness into the plaintext** before encryption. Its primary purpose is to prevent certain vulnerabilities associated with raw RSA encryption.

Padding Format:

1)The **padding begins with the byte 0x00, followed by 0x02**.
2)After these constant bytes, a random number r is inserted. This random number ensures that even if the same plaintext is encrypted multiple times, the resulting ciphertext will be different each time.
3)The padding ends with another 0x00 byte.
4)The plaintext message m is then appended to this padded format.

Usage:

Enhanced Security:
The primary purpose of this padding scheme is to enhance the security of RSA encryption. In basic RSA encryption, encrypting the same message multiple times results in identical ciphertexts, which can leak information about the plaintext.

Keeping different encryption format everytime :
By introducing randomness (r) into the plaintext before encryption, PKCS 1.5 padding prevents deterministic encryption, making it harder for attackers to identify patterns in the ciphertext.

Limitations:

Padding Oracle Attacks:
PKCS 1.5 padding is not entirely secure. One major limitation is vulnerability to padding oracle attacks, such as **Bleichenbacher's attack**. These attacks can exploit flaws in the padding scheme to recover the plaintext without prior knowledge.

Lack of Rigorous Security:
PKCS 1.5 padding was designed to add some level of security to RSA, but it does not provide the same level of security as modern padding schemes like **RSA-OAEP** (Optimal Asymmetric Encryption Padding). It relies on theoretical part rather than a rigorous mathematical security proof.

Dependant implementation:
The security of PKCS 1.5 padding also depends on proper implementation by cryptographic libraries. Inadequate implementations can introduce vulnerabilities.

Overall PKCS 1.5 padding in RSA is used to introduce randomness and prevent deterministic encryption, thereby enhancing security. However, it has some limitations and vulnerabilities, making it less secure compared to more modern padding schemes.

Dr.Shachi Natu
(Subject Teacher)

Prof. Nagaveni Hebbar
(Subject Teacher)

Roll Number:- 2105051
Name:-Niranjan Joshi
Date:- 30/09/2023

Security Lab Theory Assignment 2

Lab Outcome Attained :- LO6

What is Intrusion Detection System? Explain different types of intrusion detection systems with their working. State the advantages and limitations of each.

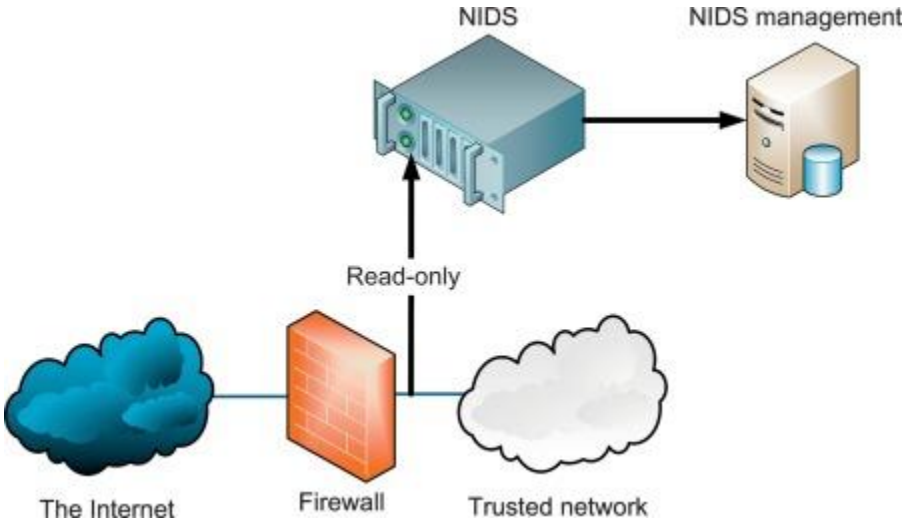
An intrusion detection system (IDS) is a system that monitors network traffic for suspicious activity and alerts when such activity is discovered.

While anomaly detection and reporting are the primary functions of an IDS, some intrusion detection systems are capable of taking actions when malicious activity or anomalous traffic is detected, including blocking traffic sent from suspicious Internet Protocol (IP) addresses.

An IDS can be contrasted with an intrusion prevention system (IPS), which monitors network packets for potentially damaging network traffic, like an IDS, but has the primary goal of preventing threats once detected, as opposed to primarily detecting and recording threats.

Types of IDS -
a)Network Intrusion Detection System(NIDS) -
A network-based intrusion detection system (NIDS) detects malicious traffic on a network. NIDS usually require promiscuous network access in order to analyze all traffic, including all unicast traffic. NIDS are passive devices that do not interfere with the traffic they monitor; Fig. 7.2 shows a typical NIDS architecture. The

NIDS sniffs the internal interface of the firewall in read-only mode and sends alerts to a NIDS Management server via a different (ie, read/write) network interface.



Working: NIDS monitors network traffic at various points within a network, such as at the network perimeter or within network segments. It analyzes packets, looking for patterns or signatures that match known attack patterns or anomalies that deviate from established network baselines. NIDS can operate in a passive or active mode, with the passive mode being more common to minimize disruption

Advantages -

1)Threat Detection: NIDS can identify and alert on a wide range of network-based threats, including malware, intrusions, and unauthorized access attempts

2)Real-time Monitoring: NIDS continuously monitor network traffic in real-time, allowing for immediate detection of threats as they occur.

3)Signature-Based Detection: Many NIDS use signature-based detection, which means they can identify known threats based on predefined patterns or signatures, making them effective at catching well-known attacks.

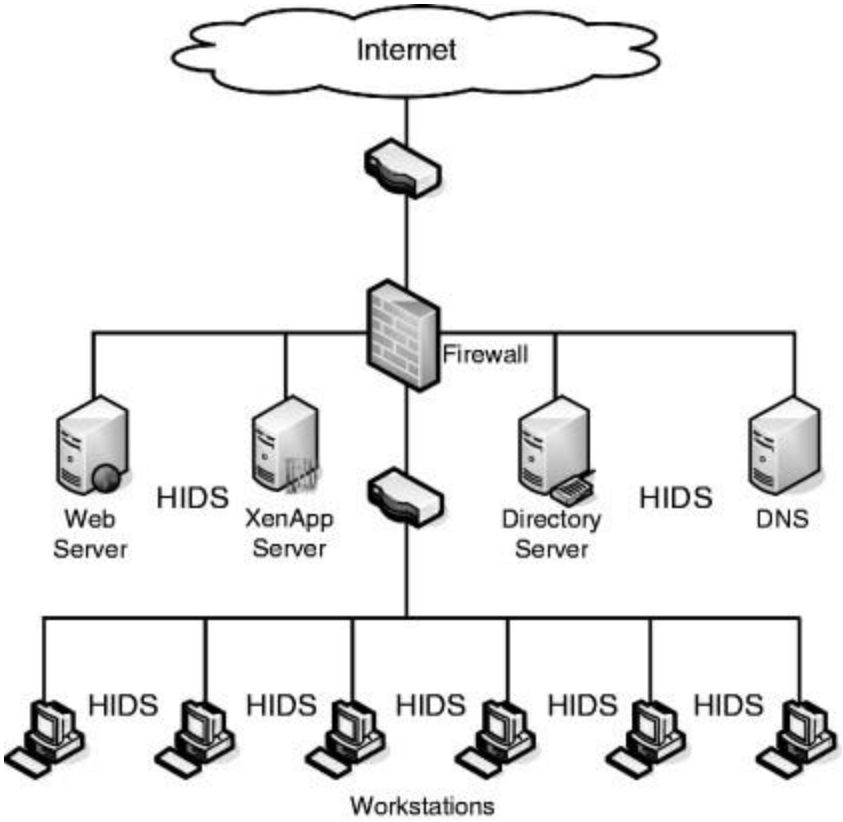
Disadvantages -

1) Inability to Detect Zero-Day Attacks: NIDS relies on known signatures or patterns to detect threats, so they may not detect new or zero-day attacks for which there are no known signatures.
2) Performance Impact: NIDS can introduce latency and impact network performance, especially in high-traffic environments.
3) Encrypted Traffic: NIDS may struggle to inspect encrypted traffic effectively, as they can't see the content of encrypted communications without decryption, which can introduce privacy and regulatory compliance concerns.

b) Host Intrusion Detection System(HIDS) -
Host-based intrusion detection systems help organisations to monitor processes and applications running on devices such as servers and workstations. HIDS tracks changes made to registry settings and critical system configuration, log and content files, alerting to any unauthorised or anomalous activity.

HIDS technologies are ‘passive’ in nature, meaning their purpose is to identify suspicious activity, not prevent it. For this reason, HIDS solutions are often used in conjunction with intrusion prevention systems (IPS), which are ‘active’.

For organisations that want to achieve deeper security visibility, host-based intrusion detection systems are commonly deployed alongside network-based intrusion detection systems (NIDS) and SIEM solutions, which aggregate and analyse security events from multiple sources.



Advantages -

1) **Host-Level Visibility:** HIDS provide detailed insight into the activities and security events occurring on a specific host, allowing for more granular monitoring and threat detection.

2) **Detection of Insider Threats:** HIDS can detect suspicious behavior from users with legitimate access to the host, making them effective at identifying insider threats.

3) **Zero-Day Threat Detection:** Unlike NIDS, HIDS can detect new or zero-day attacks because they focus on the behavior of processes and applications on the host rather than relying solely on known signatures.

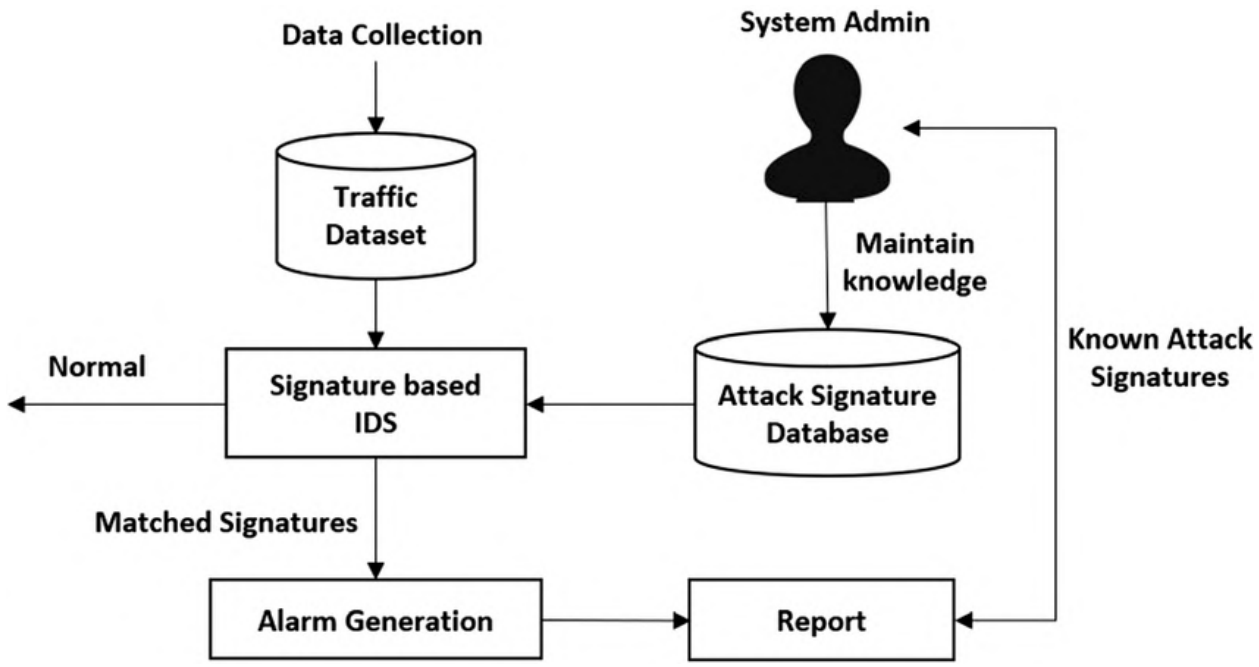
Disadvantages -

1)Limited Network Visibility: HIDS focus solely on the host they are installed on, which means they may not detect threats that occur at the network level or on other hosts.

2)Resource Consumption: HIDS can consume system resources (CPU, memory, and disk space), potentially impacting the performance of the host being monitored.
3)Complexity: Managing and maintaining HIDS on multiple hosts can be complex and time-consuming, especially in large-scale environments.

c) Signature-based intrusion detection system(SIDS) -
Signature-based detection is one of the most direct and well-established methods to identify malicious activity. Signature-based detection examines network traffic, compares it to known signatures, and generates an alert when a match is made. One example of an IDS that implements signature-based detections is Suricata. A Suricata signature consists of:

Action - what happens when the signature matches
Header - defining the network information (e.g., protocol, IP addresses, ports, and direction)
Options - defining the specifics of the rule
Signature-based detections for networks is effectively the technology that is common to antivirus and endpoint tools where a unique pattern or identifier is matched.



Advantages -

1) Effectiveness Against Known Threats: Signature-based IDS are highly effective at detecting and blocking known threats for which signatures have been created. This includes well-documented attack patterns and malware variants.

2) Low False Positive Rate: They generally have a low rate of false positives since they match network traffic against specific patterns or signatures, reducing the likelihood of alerting on benign traffic.

3) Simple Deployment: Signature-based IDS are relatively easy to deploy and configure, making them accessible to organizations with varying levels of expertise

Disadvantages -

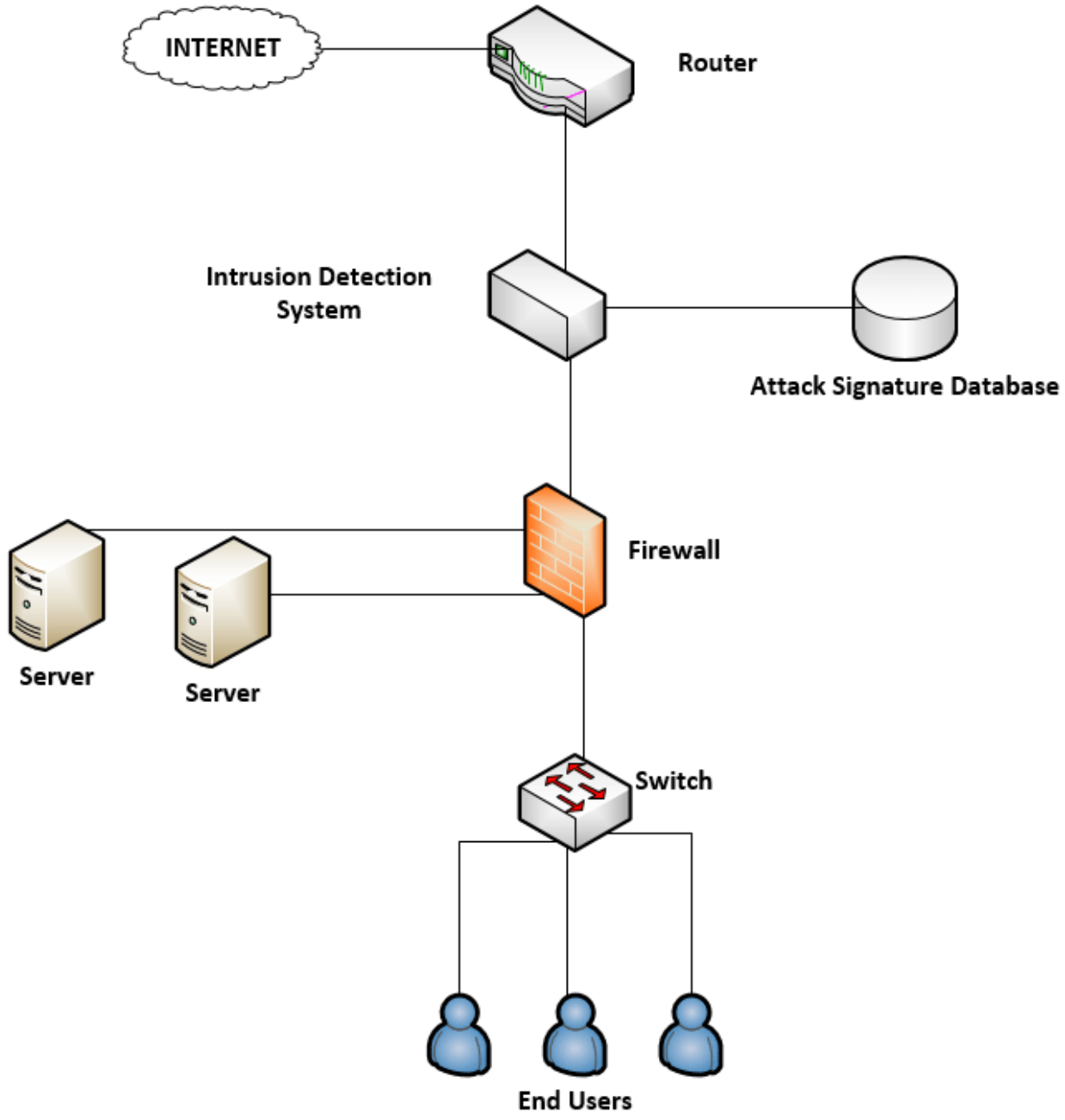
1) Ineffective Against New Threats: Signature-based IDS are less effective at detecting new or previously unknown threats (zero-day attacks) because they rely on known signatures. Attackers can easily evade detection by using novel attack techniques.

2) Limited to Known Signatures: They cannot detect attacks that do not match existing signatures, which means they might miss many advanced and evolving attack techniques.

3) False Negatives: There is a risk of false negatives if the IDS fails to recognize a known signature due to variations in the attack or the use of evasion techniques.

d) Anomaly based Intrusion Detection System (AIDS) -

Network behavior is the major parameter on which the anomaly detection systems rely upon. If the network behavior is within the predefined behavior, then the network transaction is accepted or else it triggers the alert in the anomaly detection system [3]. Acceptable network performance can be either predetermined or learned through specifications or conditions defined by the network administrator.



Advantages -

1) Effective Against Unknown Threats: Anomaly-based IDS can detect novel or zero-day attacks because they focus on deviations from normal behavior rather than relying on known attack signatures.

2) Reduced False Positives: They typically generate fewer false positives compared to signature-based IDS since they don't rely on predefined attack patterns.
3) Adaptability: Anomaly-based IDS can adapt to changing network or host environments and can detect new attack techniques or insider threats that signature-based systems might miss.

Disadvantages -
1) High False Positive Rate: While anomaly-based IDS have a lower false positive rate compared to signature-based IDS, they can still generate false alerts due to normal variations in network or host behavior, misconfigurations, or temporary spikes in traffic.
2) Baseline Creation: Creating an accurate baseline of normal behavior can be challenging, and it may take time for the IDS to learn what is truly normal, leading to false positives during the learning phase.
3) Limited Knowledge of Specific Attacks: Anomaly-based IDS may not provide specific details about the nature of an attack since they don't rely on predefined signatures. This can make it harder for security teams to understand the nature of a threat.

Dr.Shachi Natu
(Subject Teacher)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

A Survey of Moving Target Defenses for Network Security

Sailik Sengupta* Ankur Chowdhary* Abdulhakim Sabur Adel Alshamrani
Dijiang Huang Subbarao Kambhampati

***Abstract*—Network defenses based on traditional tools, techniques, and procedures (TTP) fail to account for the attacker's inherent advantage present due to the static nature of network services and configurations. To take away this asymmetric advantage, Moving Target Defense (MTD) continuously shifts the configuration of the underlying system, in turn reducing the success rate of cyberattacks. In this survey, we analyze the recent advancements made in the development of MTDs and highlight (1) how these defenses can be defined using common terminology, (2) can be made more effective with the use of artificial intelligence techniques for decision making, (3) be implemented in practice and (4) evaluated. We first define an MTD using a simple and yet general notation that captures the key aspects of such defenses. We then categorize these defenses into different sub-classes depending on *what* they move, *when* they move and *how* they move. In trying to answer the latter question, we showcase the use of domain knowledge and game-theoretic modeling can help the defender come up with effective and efficient movement strategies. Second, to understand the practicality of these defense methods, we discuss how various MTDs have been implemented and find that networking technologies such as Software Defined Networking and Network Function Virtualization act as key enablers for implementing these dynamic defenses. We then briefly highlight MTD testbeds and case-studies to aid readers who want to examine or deploy existing MTD techniques. Third, our survey categorizes proposed MTDs based on the qualitative and quantitative metrics they utilize to evaluate their effectiveness in terms of security and performance. We use well-defined metrics such as risk analysis and performance costs for qualitative evaluation and metrics based on Confidentiality, Integrity, Availability (CIA), attack representation, QoS impact, and targeted threat models for quantitative evaluation. Finally, we show that our categorization of MTDs is effective in identifying novel research areas and highlight directions for future research.**

***Index Terms*—Cyber Security, Network Security, Moving Target Defense, Artificial Intelligence, Cyber Deception, Game Theory, Attack Representation Methods (ARMs), Cyber Kill Chain (CKC), Advanced Persistent Threats, Software-Defined Networking (SDN), Network Function Virtualization (NFV), Qualitative Metrics, Quantitative Metrics, Risk Analysis, QoS Metrics**

I. Introduction

THE network and cloud infrastructures have become both ubiquitous and more complex in the past few years. Gartner predicts that by the year 2025, 80% of the entire IT infrastructure which includes deployed applications, technologies and services will be cloud-based [1]. While the performance aspects with regards to storage capacity, networking efficiency, and hardware have received due attention and evolved with business demands, aspects that govern the security of cloud infrastructure are still managed using traditional means. Given that security breaches can lead to loss of customer trust and worsen business reputation, a key question is *how effective are these traditional security approaches?* Is monitoring network traffic for malicious patterns, routinely patching known vulnerabilities, and relying on the network and perimeter defense such as Firewalls, Intrusion Detection Systems, Anti-malware tools, *etc.* enough to detect and thwart attacks?

There exist multiple shortcomings in these traditional defense mechanisms. First, the attacker, with time on their side, can spend reconnaissance effort in modeling the cloud system (the defenses in place) and then, carefully plan their attacks. Second, implementations of these defenses in practice are often far from ideal, thereby allowing attackers even more opportunities to exploit the system. A report from 2016 predicts that by the end of 2020, 99% of the vulnerabilities exploited will be known to security and IT professionals since a year ago [2]. A major reason for this is the time and complexity associated with the process of routinely patching vulnerabilities. Often, the fear of degradation in the Quality of Service (QoS) provided to customers deters Cloud Service Providers (CSPs) from making changes to the existing system configuration. Third, *zero-day* attacks developed using the information the attacker gathers during the reconnaissance phase can render traditional defenses useless.

To address the shortcomings of existing defense methods, *Moving Target Defense* (MTD) has emerged as a solution that provides *proactive defense* against adaptive adversaries. The goal of MTD is to constantly move between multiple configurations in a cyber-system (such as changing the open network ports, network configuration, software, *etc.*.) thereby increasing the uncertainty for the attacker; in effect, diminishing the advantage of reconnaissance that an attacker inherently has against traditional defense mechanisms. The advantages of MTDs go away if the shifting mechanism is deterministic because the attacker, with time on their side, will eventually be able to predict this movement and design attacks accordingly. Thus, for MTDs to be effective, they need to have implicit randomness built into them. This survey categorizes MTDs based on what they shift, when they shift and how they shift.

The dynamic aspect of MTD adds an extra layer of complexity in implementing these defenses. To address this,

*These two authors have contributed equally to this work.

The authors Sailik Sengupta, Ankur Chowdhary, Abdulhakim Sabur, Dijiang Huang, and Subbarao Kambhampati are with the School of Computing, Informatics, and Decision Systems Engineering (CIDSE), Arizona State University, Tempe, AZ, USA (e-mail: ssengu15@asu.edu, achaud16@asu.edu, asabur@asu.edu, dijiang@asu.edu, rao@asu.edu). Abdulhakim Sabur is also with Taibah University, Madinah, Saudi Arabia. Adel Alshamrani is with University of Jeddah, Jeddah, Saudi Arabia (e-mail: asalshamrani@uj.edu.sa)

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

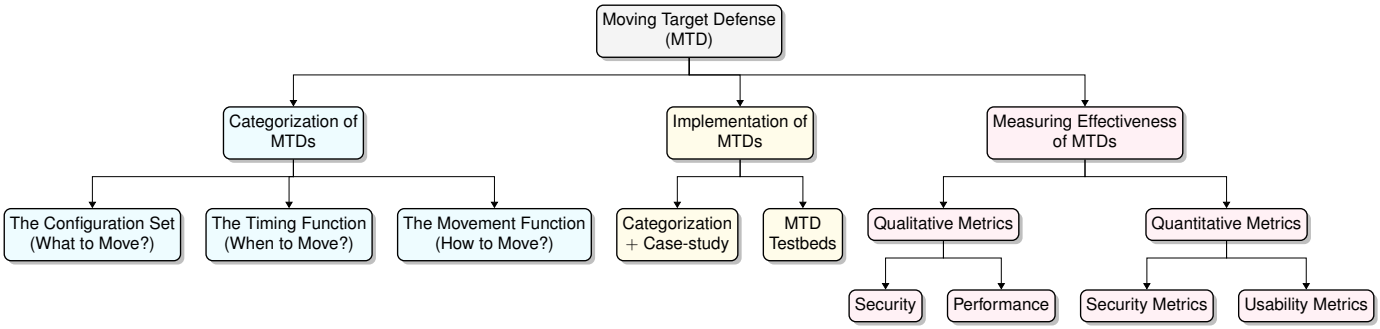


Fig. 1: We survey various defense techniques based on the paradigm of Moving Target Defense (MTD) and showcase that a common terminology and directions for future work naturally emerges due to the categorization in our survey.

one can leverage advances in networking technology. First, *Network Functions Virtualization* (NFV) [3] has emerged as a technology to provide a virtualized implementation of hardware-based equipment such as firewalls, routers, *etc.*. through *Virtual Machines* (VMs) or containers running on top of a physical server in cloud computing environments. Given MTDs, often, need more hardware than conventional software systems, such virtualization helps to reduce the cost of implementation. Second, *Software-Defined Networking* (SDN) [4], which serves as an enabling technology for NFV, provides a centralized security policy enforcer. The programmable interface afforded by SDN can help administrators implement optimal movement strategies for MTDs. Futhermore, enabling MTD implementation at scale will help us evaluate the effectiveness of these defenses in practical settings.

The key contributions of this survey are: (1) provides an umbrella under which we can categorize the array of MTD techniques proposed for network environments, (2) introduces a common language that can be utilized to describe (and understand) the assumptions and threat models of various MTDs, (3) gives an overview of how these defenses are implemented by researchers, highlighting testbeds and case studies to guide its large-scale deployment and (4) discusses how MTDs have been evaluated from a qualitative and a quantitative standpoint, in effect, shedding light on how effective these tools and techniques are with regards to security and performance. Figure 1 gives a a quick overview of this survey.

The rest of the survey is organized in the following manner. In Section II, we introduce the reader to some background knowledge about the various stages of an attack in cloud systems, popular for detections and defenses against malicious traffic, and formal frameworks to capture knowledge about attacks in networking systems. In Section III, we propose a universal notation that captures the key aspect of all the MTDs proposed and use it to investigate and categorize the defenses depending on how they answer the questions (1) *what to move*, (2) *when to move* and (3) *how to move*. In this regard, we also highlight how the different cyber surfaces - discussed under *what to move*– relate to the various stages of an attack described in Section II. In Section IV, we discuss how the various MTD works have been implemented in practice, with special emphasis on the role of (and the effectiveness) of SDN and NFV in enabling them. We showcase examples of existing MTD testbeds and perform case-studies that can help security personnel in the adoption of MTD solutions for production-grade networks. In Section V, we elaborate on various qualitative and quantitative metrics that have been presented in the literature; this can help a cloud administrator decide if a defense mechanism is secure enough. In Section VI, we highlight areas, in terms of our categorization, that have received less attention and discuss an array of future research directions. Finally, we conclude the survey in Section VII.

II. Background and Related Work

In this section, we first discuss related surveys in the area of Moving Target Defense (MTD) that look at the aspects of characterizing and evaluating proactive defenses. This helps us to situate our survey and highlight several aspects of MTDs that existing surveys ignore. Second, we describe the various stages of an attack that establishes a realistic threat model against a cyber-system. This helps us understand which step(s) of an attack a particular MTD technique seeks to disarm. Third, we highlight the traditional defense methods that are presently used to detect or reduce the impact of cyberattacks. As seen later, MTD mechanisms often leverage these traditional defenses, adding movement to the way they are deployed; this makes it harder for an attacker to fool the overall defense. Finally, we provide an overview of existing databases (NVD, OVSDB, CVSS) [11], [12] that are used to obtain domain knowledge about known vulnerabilities and popular attack representation methods such as attack graphs and attack trees.

A. Related Works and the Need for this Survey

We present a comparison of our survey to existing surveys in Table I. Firstly, we observe that most existing surveys [6], [9] provide only partial coverage of topics relating to *what, when, and how* to move the elements of the network. Section III provides a more holistic view of various Moving Target Defenses (MTDs). Moreover, the techniques surveyed do not talk about modeling Advanced Persistent Threat (APT) scenarios [13], [14]. Our survey, on the other hand, provides an overview of APT and its relation to the attack and defense surfaces, thereby helping us to highlight how a particular MTD may be effective against both known attacks as well as unknown attacks.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

Research Work	Cyber Surface Shift Analysis			Relation to APTs	MTD Implementation	MTD Evaluation		Research Directions
	What?	When?	How?			Qualitative	Quantitative	
[5]	✓	✓	✗	✗	✗	✗	✗	✓
[6]	✓	✗	✓	✗	✗	✗	✗	✗
[7]	✓	✗	✗	✗	✗	✓	✗	✓
[8]	✓	✗	✗	✗	✗	✗	✓	✗
[9]	✓	✗	✓	✗	✗	✓	✓	✓
[10]	✓	✗	✗	✗	✗	✓	✗	✓
Our Survey	✓	✓	✓	✓	✓	✓	✓	✓

TABLE I: Our categorization of MTDs is a super-set of the categorization considered in earlier works. The analysis of MTDs in the context of Advanced Persistent Threats (APTs), a unified view to analyse the what, when and hows of MTDs and case study of their implementation are entirely new.

We provide an in-depth analysis of how MTDs are implemented and the role of networking technologies such as SDN and NFV in enabling them. We categorize the MTDs based on the maturity level of their implementation–ranging from simulation-based analysis to research test-beds and production-level industrial products. Table V summarizes various MTDs highlighting their use of centralized networking paradigms such as SDN/NFV (yes/no), and the level of maturity at which they have been implemented (analytic/simulation/emulation/commercial). This categorization has not been considered in previous surveys.

Some existing surveys [8], [9] have taken a look at categorizing the evaluation metrics for understanding the effectiveness of MTDs but these works do not talk about the different components that contribute to the evaluation of MTDs. In our analysis, we consider both security and performance metrics associated with each system configuration and with the ensemble, enabling us to highlight directions that can help improve existing MTDs solutions and mechanisms.

Beyond simply providing a categorization of existing work, the goal of our survey is to establish a common language for researchers who develop MTDs. This will help in making evident the aspects that have been considered and those that have been assumed away in the development of a particular MTD in the future. Our categorization also helps to identify promising directions for future research such as prevention, and hybrid surface shifting, improving the modeling of APTs, *etc.* (see Figure 14).

B. Attack Modeling Techniques

Organizations utilize advanced infrastructure management tools and follow best practices such as software patching, hardening, analysis of the system's log for reducing the attack surface. Yet, skilled adversaries manage to compromise the network assets by utilizing zero-day attacks, customized malware, *etc.*, that are often difficult to detect or prevent using intrusion detection systems and anti-virus tools. For the effective deployment of intelligent cyber-defenses, it is crucial to collect information (called the threat model) about the attack process followed by an adversary.

An intelligence-driven approach focused on studying particular threats from an attacker's perspective is key for the detection and mitigation of sophisticated attacks in networks, which are characterized as *Advanced Persistent Threats* (APTs) [14], [13]. To understand the collection, correlation and categorization of data related to a cyberattack, Lockheed Martin defines the *Cyber Kill Chain* (CKC) [15]. The evidence-based knowledge derived from their study can help us in understanding and deploying appropriate defense measures. Thus, we will first describe the different phases of the CKC followed by a brief description of APTs and how they can be viewed through the lens of CKC. This setup will later help us understand how MTDs can be effective against the different phases of an APT.

1) Reconnaissance: The attacker gathers information about the target environment in this phase. For example, the attacker can perform passive monitoring using automated tools such as trace-route and Nmap to perform network probes.

2) Weaponization: The attacker, based on information obtained in the reconnaissance phase, utilizes tools and techniques such as a phishing e-mail, a malware-infected document, *etc.* to create a targeted attack payload against the victim.

3) Delivery: The transmission of infected payload occurs during this stage. For example, the attacker may leave a malware-infected USB on the victim site or send an email to the Chief Financial Officer (CFO) of the company. This step requires the attacker to evade the authentication and thus, the people (and not the technology) become a more important target during this phase. Thus, a trained workforce can help in reducing the attack surface area.

4) Exploitation: The attack detonation takes place during this stage. This phase involves the exploitation of a vulnerability and the attacker gaining elevated privileges on the victim's resources by utilizing specially crafted payload that exploits a known (or zero-day) vulnerability.

5) Installation: Once the attacker gains elevated privileges in the exploitation stage, they may install malware on the victim's machine or choose to harvest useful information in the victim's database. Tools that can analyze abnormal behavior such as anti-malware, *host-based IDS* (HIDS), *etc.* become quite important in attack detection during this stage.

6) Command and Control (C&C): After the installation phase is complete, the attacker contacts the $C\&C$ to maintain remote control over the victim machine. Tools such as *network-based IDS* (NIDS) and outbound *firewall rules* are quite useful in detecting and blocking malicious outbound connections requests.

7) Actions on Objectives: During this phase, the attacker executes the actions to achieve the attack goals, such as data-exfiltration, service disruption, *etc.* Two other important behaviors often observed in this attack-step are *pivoting*, which involves identifying similar target nodes that have already been

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

exploited, and *lateral movement*, which involves identifying new systems that can be exploited in the network.

C. Advanced Persistent Threats (APTs)

Advanced Persistent Threats (APTs) refers to a distinct set of attacks against a high-value target organization that differs from normal cyber attacks in several ways [13]. First, APTs are achieved by a group of highly-skilled attackers who are well-funded. According to *Mandiant Report* [16], APTs such as *Operation Aurora, Shady Rat, and Red October* have been used on a global scale for industrial espionage [17], [18]. Oftentimes, the attackers mounting APT work closely with government organizations. Second, an APT attacker is extremely persistent; they (1) pursue the objectives repeatedly, often over an extended period, (2) can adapt to a defender's efforts in trying to resist the attack, and (3) determined to maintain a level of access within the defender's system required to achieve their objectives. Third, the key objective of APTs is to either exfiltrate information or undermining the key services in the network using multiple attack vectors [19].

a) Relation between APTs and CKC: An APT can be broken down into five phases– reconnaissance, foothold establishment, lateral movement, data exfiltration, and post exfiltration [13]. These can be mapped to different phases of the Cyber Kill Chain. The *reconnaissance* phase in APTs maps directly to the *reconnaissance* phase in CKCs, which was described earlier. The *foothold establishment* phase comprises of the *weaponization* and the *delivery* phases of CKC. The attackers use information gathered from the reconnaissance phase in order to prepare an attack plan and eventually exploit vulnerabilities uncovered in the target organization's web application, databases, and other software.

Once the attacker has gained a foothold into the target environment, they can try to move laterally in the target environment and gain access to other sensitive hosts and critical information in the organizational network in the lateral movement phase of APT. In this setting, the attacker needs to continuously explore (perform reconnaissance) and exploit the various components of the defender's system, mapping to the exploitation phase of the CKC.

In the data ex-filtration phase of APT, the attacker tries to exfiltrate the collected data to their *command and control* $(C\&C)$ center. Most of the intrusion detection and prevention systems do ingress filtering and not egress filtering, thus, data exfiltration can often go undetected. The goal of the attacker in *post ex-filtration* phase is to maintain persistence in the system for a long duration of time until the funding for attack campaign is finished or the goals of attacking the organization/government are fulfilled.

D. Defense Methods

In this section, we provide a brief overview of the various defense techniques and highlight how each of these defense mechanisms is effective in various parts of a Cyber Kill Chain (CKC) in Table II. This discussion will help the reader better understand some of the MTD techniques that use these traditional defenses as a building block.

Phase	**Network Defense Techniques**				
	Detect	**Deny**	**Disrupt**	**Degrade**	**Deceive**
Reconnaissance	Web Analytics				
Weaponization	NIDS	NIPS			
Delivery	Vigilant user	Proxy filter	AV	Queuing	
Exploitation	HIDS	Patch	DEP		
Installation	HIDS	chroot	AV		
C2	NIDS	ACL	NIPS	Tarpit	DNS redirect
Actions on Objectives	Audit log			QoS	Honeypot

TABLE II: Highlights how existing defense mechanisms are related to the different phases of the Cyber Kill Chain.

1) Web Analytics: A huge amount of security-related information is present on the web– in *social engineering* websites, phishing sites, and dark-web forums– including discussion about a particular target product or company. As discussed by Glass *et. al.* [20], the problem of exploring and analyzing the web for information should provide (1) Security relevant information discovery capabilities (2) *Situation awareness* by performing real-time inference over the available information, and (3) predictive analysis to provide early warning for any likely security attacks.

2) Intrusion Detection Systems (IDS): There are two types of IDS agents that can be deployed in the network under attack, i.e., Network-based IDS (NIDS) and Host-based IDS (HIDS). NIDS such as *Bro* [21] and *Snort* [22], use signature match techniques based on known attack patterns and can flag incoming network traffic as malicious or benign. HIDS such as *auditd* [23] or *tripwire* [24], on the other hand check the Indicators of Compromise (IOCs) on the physical server or VM, in order to identify malicious activity, e.g., privilege escalation attempt by normal user.

3) Intrusion Prevention Systems (IPS): A network security threat prevention technology such as IPS [25] examines network traffic flow to detect and prevent vulnerability exploits. The exploits come in the form of malicious inputs to the target application or service. The IPS is often deployed behind the firewall and provides a complementary layer of analysis. Compared to IDS, the IPS system takes automated action based on traffic pattern match such as (1) dropping malicious traffic packets, (2) blocking requests from a source, (3) resetting connection, and (4) sending an alarm to the administrator.

4) Proxy Filtering: A (reverse) proxy server such as *nginx*, can act as an intermediary between the attacker located on the public network and private network. A proxy can help in shielding the real network from the attacker.

5) Access Control List (ACL): ACL can be applied at different enforcement points in a network. ACLs, such as *netfilter* [26], investigate network traffic and based on properties such as source/destination IP address, port number, protocol *etc.* decide either to *permit* or *deny* it.

6) Data Execution Prevention (DEP): DEP is a security feature that can be deployed in the form of hardware or software to prevent malicious code from running on the host. They *monitor programs* run on the host and ensure it uses memory in an expected (or safe) manner.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

7) Anti-Virus (AV): A software program designed to protect the hosts from malicious programs such as a virus, spyware, worms, rootkits, key-loggers, *etc.* The AVs can be classified into two types– *signature-based AV* and *behavior-based AV*. The signature-based AVs check the malicious program against the database of known malicious programs. On the other hand, the behavior-based AVs check the program behavior by running it in a sandbox environment.

8) Tarpit: This defense technique involves purposeful introduction of delay when responding to queries. The key idea behind this defense mechanism is that adversaries will give up on a target if it takes too long to achieve the defined goal.

9) QoS: The *network traffic* can be segmented on the service type and importance. The segmented traffic can be analyzed for bottlenecks and threats. The malicious traffic can be slowed down in order to increase the *Cost of Attack (COA)* or selectively dropped.

10) DNS Redirect: The malicious connection requests can be served a different response than was expected. The attacker may seek to connect with command and control center using a page with malware, but DNS redirect will kill this chance.

11) Honeypot: A security mechanism, which can be used to detect, deceive or in some cases counter a malicious user trying to gain access to key network services [27]. Honeypots such as *Cowrie* [28] can help in better understanding the attacker's tools and tactics. The connection requests from the attacker are directed to a decoy service, which mimics the behavior of the normal service and logs the attacker's activity.

Although there exists a large set of defense mechanisms, attackers often use clever techniques to evade detection or prevention. SANS [29] discusses methods like *obfuscation*, *fragmentation*, *encryption* and *Denial of Service* (DoS) attacks against signature-based detection tools. Detection based on HIDS can also be evaded by a skilled attacker using techniques such as file location manipulation (using directories or files white-listed by IDS), application hijacking, *etc.* Furthermore, machine learning models that can help overcome some of the shortcomings of signature-based detection tools can themselves be fooled by adversarial attacks [30], [31]. On similar lines, deception techniques such as DNS redirect and Honeypot can help in deceiving the attacker temporarily, but over a longer period of time, the attacker will eventually change their attack methodologies thereby reducing the effectiveness of these defenses. Stojanovski *et. al.* [32] performed experimental analysis on how to bypass *DEP protection* on *Windows XP* systems. Thus, there is a need to perform intelligent manipulation to assure the attacker's likelihood of reaching the desired goal is limited, even if they can evade the traditional detection methods. Additionally, the defense mechanisms discussed in the Section II-D target known attacks often with easy to detect signature patterns.

a) Proactive Defenses against APTs: The differences between APTs and regular cyberattacks, discussed in II-C, make it arduous to use traditional defenses and pre-specified threat models to address APTs as a whole. In this regard, proactive defenses can prove to be effective against APTs. While MTDs, as we will see later, makes it difficult for APT attackers by dynamically shuffling various system components (see Fig 3), other proactive defenses such as cyber-deception can prove to be effective in gathering threat-model information. For example, Shu *et al.* propose a cyber deception to protect *FTP* services against APT attackers [33]. In their research work, a defender reroutes attack traffic to a host, which may be a honeypot, and the defender ensures that an attacker is not able to notice a connection difference between the real IP address and the honeypot trap. By observing attacker's behavior on the honeypot, the defender updates the threat-model and in turn, hardens their FTP services. A key aspect of this work is to make the attackers continuously believe that they are interacting with the original environment as opposed to a honeypot.

E. Domain Information for Modeling Cyber-attacks

Defenders almost always have information about the system they want to protect. This knowledge can help in enhancing the threat-model and, in turn, improve the effectiveness of MTD techniques. Such information may range from knowledge of the network architecture, the capacity of the individual machines, known vulnerabilities (and an idea of how dangerous they are), *etc.* We discuss here a few popular models and data sources that have been leveraged by researchers.

1) Common Vulnerabilities and Exposures (CVEs): are publicly known vulnerabilities and exposures that are categorized and can be referenced uniquely in the *National Vulnerability Database* (NVD) using a common vulnerability enumeration identifier (CVE-ID). This system is maintained and updated by the *Mitre* corporation regularly to make defenders and administrators aware of existing and new vulnerabilities.

2) Common Vulnerability Scoring System (CVSS): The use of the Common Vulnerability Scoring System (CVSS) for rating attacks is well studied in security [34]. For (most) CVEs listed in the NVD database, we have a six-dimensional *CVSS v2* vector [12], which can be decomposed into multiple components that represent Access Complexity (AC), i.e. how difficult it is to exploit a vulnerability, and the impact on Confidentiality, Integrity, and Availability (CIA) gained by exploiting a vulnerability. The values of AC are categorical $\{$EASY, MEDIUM, HIGH$\}$, while CIA values are in the range $[0, 10]$. These scores are also known as the Exploitability Score (ES) and the Impact Score (IS).

Although a defender may be aware of the known vulnerabilities present in their system (by being aware of the published CVEs that affect them), the knowledge of how they affect their system, in particular, may help them in making better decisions for security. The attack representation can be useful to quantify the attack and defense surface for MTD. To this extent, we define two heavily used representation methods that can represent known attacks present in a system– the *attack tree* and the *attack graph* as shown in the Figure 2.

The Figure 2, shows a network attack scenario, where an attacker has access to FTP and SSH server over the network. This example illustrates different methods for representing multi-stage attacks, i.e., attack graph and attack tree which we defined above. The FTP server consists of a vulnerability that allows the attacker to obtain remote shell (rsh) on the

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

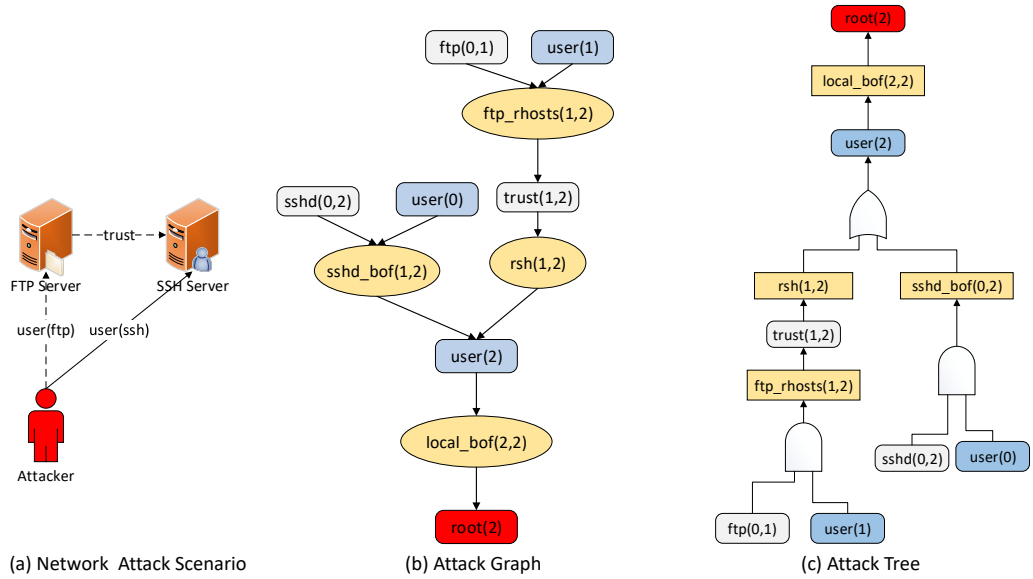


Fig. 2: Attack representations such as an attack graph (b) and an attack tree (c) for a simple network attack scenario (a) in a small-size cloud system with known vulnerabilities. Creation of these representations from network vulnerability and reachability information often suffers from scalability issues for real-world networks (see Table III).

system. The SSH server, on the other hand, consists of buffer overflow ($sshd_bof$) vulnerability. The goal of the attacker is to obtain root privilege on the SSH server, i.e., *root(2)*. The attack progression using attack graph and attack tree has been presented in Figure 2 (b), (c) respectively.

Attack Tree [35] as shown in Figure 2(c) is another method of representing system security. The Attack Tree represents the network attacks. Attack Tree represents a monotonic path taken by an attacker starting from a leaf node to reach a goal node. Attack Tree usually consists of set of *AND* nodes (sshd(0,2), user(0) - Figure 2(c)) and *OR* nodes (rsh(1,2), $sshd_bof$(0,2)). The *OR* nodes represent one or more ways in which a goal node can be reached, whereas *AND* nodes represent different conditions that must be fulfilled to achieve a goal node. Children of the node are refinements of this goal, and the attacks that can no longer be refined are represented by leaf nodes [36].

Definition 1. *An Attack Tree [36] can be defined by three tuples* $(N, \rightarrow, N_r)$

- *N is all possible nodes in the tree;*
- $S^+(N)$ *is* multi-set *of all possible subsets of nodes N;*
- $\rightarrow \subseteq N \times S^+(N)$ *denotes transition relation;*
- N_R *represents the* goal node *of the attack tree.*

Attack Graph is a data structure used to represent attack propagation in a network with vulnerabilities as shown in Figure 2(b). The start state of the attack graph represents the current privileges of the attacker. The goal state of the attack graph represents a state in which the intruder has successfully achieved his attack goal, e.g., data-exfiltration, root privileges on a web server, *etc.* Security analysts use attack graph for attack detection, network defense, and forensics [45]. We formally define the attack graph as follows:

Definition 2. *Attack Graph (AG) An attack graph is represented as a graph* $G = \{V, E\}$*, where V is set of nodes and E is set of edges of the graph G, where*

1) $V = N_C \cup N_D \cup N_R$*, where* N_C *denotes the set of conjunctive or exploit nodes,* N_D *is a set of disjunctive nodes or result of an exploit, and* N_R *is the set of a starting nodes of an attack graph, i.e. root nodes.*
2) $E = E_{pre} \cup E_{post}$ *are sets of directed edges, such that* $e \in E_{pre} \subseteq N_D \times N_C$*, i.e.,* N_C *must be satisfied to obtain* N_D*. An edge* $e \in E_{post} \subseteq N_C \times N_D$ *means that condition* N_C *leads to the consequence* N_D*.* E_{pre} *represents the attack graph pre-conditions (ftp(0,1) and user(1) in the Figure 2(b)) necessary for vulnerability exploitation and* E_{post} *are the post-conditions (rsh(1,2) in the Figure 2(b)) obtained as a result of exploit.*

The time taken to construct attack representation methods (ARMs) grows exponentially with an increase in the number of hosts or the number of known vulnerabilities in the network [46]. Authors in [47] survey various research works that try to address this challenge. Amman *et al.* [48] present a scalable solution that assumes monotonicity; it allowed them to achieve scalability of $O(N^6)$. To mitigate the state explosion problem, most of the existing solutions try to reduce the dependency among vulnerabilities by using some logical representation [49] or by imposing a hierarchical structure to reduce the computing and analysis complexity of constructing and using ARMs [50]. In the latter work, the authors proposed a two-layer AG generation approach where the goal was to develop a faster method by considering network reachability and vulnerability information at different layers. The graphs constructed have vulnerability information of the individual hosts in the lower graph while the topological information of the network

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

Category	Details	Complexity
Automated Attack Analysis [37]	Multi-prerequisite graph based on vulnerability and reachability information	O(E+NlgN); N is attack graph nodes and E is graph edges
Attack Cost Modeling [38]	Time Efficient Cost Effective hardening of network using Attack Graph	$O(n^{\frac{d}{2}})$; d represents the depth of the attack graph
Attack Cost Modeling [39]	Model checking based attack graph generation using Markov Decision Process (MDP)	Approximation algorithm $\rho(n) = H(max_{a \in A} \{\mu_G(a)\})$, where A is Attacks, μ is maximization function.
Scalable Attack Graph [40]	Scalable attack graph using logical dependencies.	$O(N^2) - O(N^3)$, where N is number of nodes in attack graph.
Attack Graph based Risk Analysis [41]	Scalable attack graph for risk assessment using divide and conquer approach	$O(r(n+c)^k)$, where r is small coefficient.
Attack Cost Modeling [42]	Attack Graph cost reduction and security problem solving framework Min. Cost SAT Solving.	NP-Hard problem, SAT solving methods employed.
Ranking Attack Graphs [43]	Asset Ranking algorithm for ranking attack graphs to identify attacks. *Page Rank* based algorithm	Similar to complexity of page rank algorithm.
Attack Cost Modeling [44]	Identifying critical portions of attack graph. Min. Cost SAT solving, Counter-example guided abstraction refinement (CEGAR)	NP-Hard problem, SAT solving methods used.

TABLE III: Complexity of the various Attack Representation methods.

is in the upper layer. Unfortunately, the effectiveness of these methods on real-world networks is uncertain.

Table III highlights the complexity of creating various attack graph and attack tree-based methods. As can be see, scalability is a major concern when coming up with a good attack representation, thereby impacting the effectiveness of MTD techniques. In [51], the authors present a scalable solution for approximating the attack graph of a large scale data-centric network by using distributed attack graph computation followed by semantic clustering. We discuss attack representation methods as one of the *Quantitative metrics* which can be used for measuring effectiveness of MTD in Section V.

III. Categorizing Moving Target Defenses

The goal of *Moving Target Defense* (MTD) is to continually move the components of an underlying system in a randomized fashion; this ensures the information gathered by the attacker in the *reconnaissance phase* becomes stale during the *attack phase* given the defender has moved to a new configuration within that time. This increases the uncertainty for the attacker, making it difficult (or rather, more expensive) for them to successfully exploit the system.

An MTD can be described using a three tuple $\langle M, T, C \rangle$ where M represents the movement strategy that directs the system *how to move*, T denotes the timing function that represents *when* a switch action occurs in the MTD system and C represents the configuration space or the set of configurations between which the system switches, answering the question of *what to switch*. Answers to these three questions can help us define a useful categorization for the various MTDs. We believe that this categorization (1) gives a holistic view of the various MTD systems, providing a heuristic sense of what modeling aspects, when done carefully, make a particular defense effective, and (2) highlights opportunities on how modeling of a particular component of an MTD can either be improved or amalgamated with other MTDs.

As we will see, the configuration set C and the timing function T are often a design choice made the system administrator depending on the threat model. However, the movement function $M : H \rightarrow C$ needs to be a stochastic function that given the history of system configurations H as input, produces the next configuration the system should switch to. A stochastic function is needed because a deterministic function can easily be learned by an attacker over time and thus, does not help the defender in terms of security.

A. The Configuration Set C – What to Switch?

At an abstract level, from the perspective of an attacker, every software system can be categorized into four surfaces:

- Exploration Surface
- Attack Surface
- Detection Surface
- Prevention Surface

In this section, we use these surfaces as a basis for categorizing what the different MTDs shift. In the context of multi-stage attacks like APTs, an adversary needs to exploit all the different surfaces, but not necessarily in a predefined order (see Fig. 3). For example, an adversary may first try to explore the target network and try to figure out its topology, bandwidth, software deployed on the different nodes, *etc.*, but it may also need to perform reconnaissance at an advanced stage, say, to verify if it has actually gained access and established a foothold, i.e. a new vantage point, within the system. The knowledge gained in the exploration phase helps them to execute attacks that exploit the system, to move to different points in the network, and exfiltrate important information. As both exploration and attack traffic are malicious in nature, they tend to differ from legitimate user traffic. At that point, the detection and prevention surfaces come into play.

In Fig. 4, we show a Venn diagram that categorizes existing MTDs based on the surface they shift. Although this categorization is at a level of abstraction, we discuss various works and show how different MTDs define the elements of the set C. Most MTDs shift one surface at a time, rarely considering scenarios where other surfaces can be shifted in conjunction.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

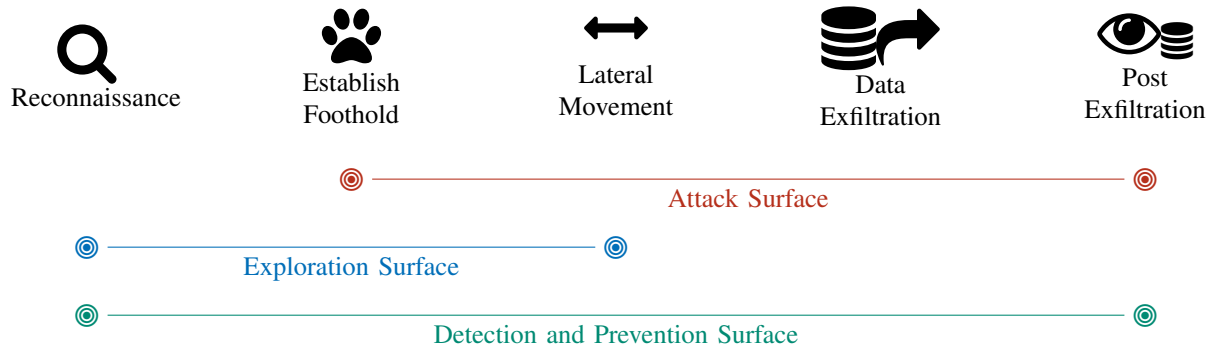


Fig. 3: A mapping between the different phases of Advanced Persistent Threat (APT) and various surfaces of a cyber system that various MTDs seek to move. Shifting of the exploration surface and the attack surface are effective only against some phases on an APT, whereas shifting the detection and the prevention surface is effective throughout the APT life-cycle.

After discussing the various MTDs, we briefly mention some unexplored research areas that can lead to the development of new defenses that shift multiple surfaces.

*1) **Exploration Surface Shifting**:* The goal of shifting the exploration surface is to ensure that the model of a system that an attacker can gather by exploration actions such as by scanning for open-ports, sending non-malicious traffic to uncover system topology, discover vulnerabilities, *etc.*, are noisy or inaccurate. Thus, an adversary, with this faulty information from the reconnaissance phase, is left with no other choice but to work with faulty view of the attack surface.

In [52], [53], the authors propose the concept of Random Host Mutation (RHM) – moving target hosts are assigned random virtual IP addresses (vIP) in an unpredictable and decentralized fashion. Formally, C represents the set of bipartite graphs, where each configuration represents a mapping from the set of vIP addresses in the Virtual Address Range (VAR) to the set of hosts in the network. Every switch action changes the one-to-one mapping of hosts in the system to VARs. Another work [54] tries to implement an MTD on the same set of configurations using a centralized approach based on *(SDN)* technologies like *OpenFlow*.

Another line of work focuses on reducing the quality of information that an attacker can gather through exploration. In [55], the authors use a centralized approach to obfuscate the network links so that the topology that an attacker can retrieve via crossfire attacks is noisy and unreliable. In this work, each state is a possible path from the source to the destination of the crossfire attack. Thus, the configuration space C represents the set of all paths from a source point to a destination point in the network. The MTD paradigm comes into play because the administrator chooses a path, in a randomized fashion, so that the attacker is not able to get reliable path information between any any two points in the network. Given attacks may only succeed if the packet is sent over a particular path, attackers are forced to use a lot of attack traffic in order to succeed in exploiting the system or even obtain a reasonable estimate of the network topology. Such behavior increases their chances of getting caught or dealing with an inaccurate (and hopefully, not useful) model of the system. On similar lines, there have been works that either try to move the fingerprint of a protected host [56] or randomly alter the time schedule that guides when a host transmits information, reducing the effectiveness of selective jamming attacks against smart meters [59]. In [57], the authors propose to send back incorrect information to an attacker trying to query information about the hosts on the network. Although they come up with deterministic strategies for responses, the possibility of sending back different lies opens up when the underlying attack surface uses an MTD. These works are termed as *cyber-deception* because the defender is trying to deceive the adversary by feeding them false information about the system. In such cases, the goal of the defender is to ensure that the adversary's model of the underlying network is incorrect as opposed to just being incomplete or noisy.

In [58], Jajodia et. al. looks at how they can deploy *honey-nets* in a strategic fashion and thereby, minimize the expense of deploying all possible honey-nets at once. They show that deploying honey-nets introduces deception in the network against *noisy-rich* (NR) cyber-attackers (i.e. adversaries who try to exploit all the vulnerabilities present in order to compromise the target network). In this case, if we represent the set of all honey-nets that can be placed in a network as X, then the configuration space C consists of all budget-limited subsets of honey-nets that can be deployed by the administrator.

*2) **Attack Surface Shifting**:* Most of the work from the research community has focused on movement of the attack surface. The main aim of switching between attack surfaces is to render invalid an attack action that the attacker chooses after some exportation. For example, an attack to exploit a *Linux-based OS* will be useless if it is executed on a machine running a *Windows OS*. We first discuss some MTD methods that are, for a networking environment, defined at an abstract level. We will then, focus on ones that consider moving more specific elements of the network.

In one of the earlier works [62], the authors have a set of systems, which forms their space of MTD configurations C, and each configuration $c \in C$ can be represented by an attack surface characterized by three properties– (1) the set of entry and exit points into the system, (2) the set of *channels*, and (3) the set of un-trusted items (data/system/network link). The defender aims to switch between the different configurations

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

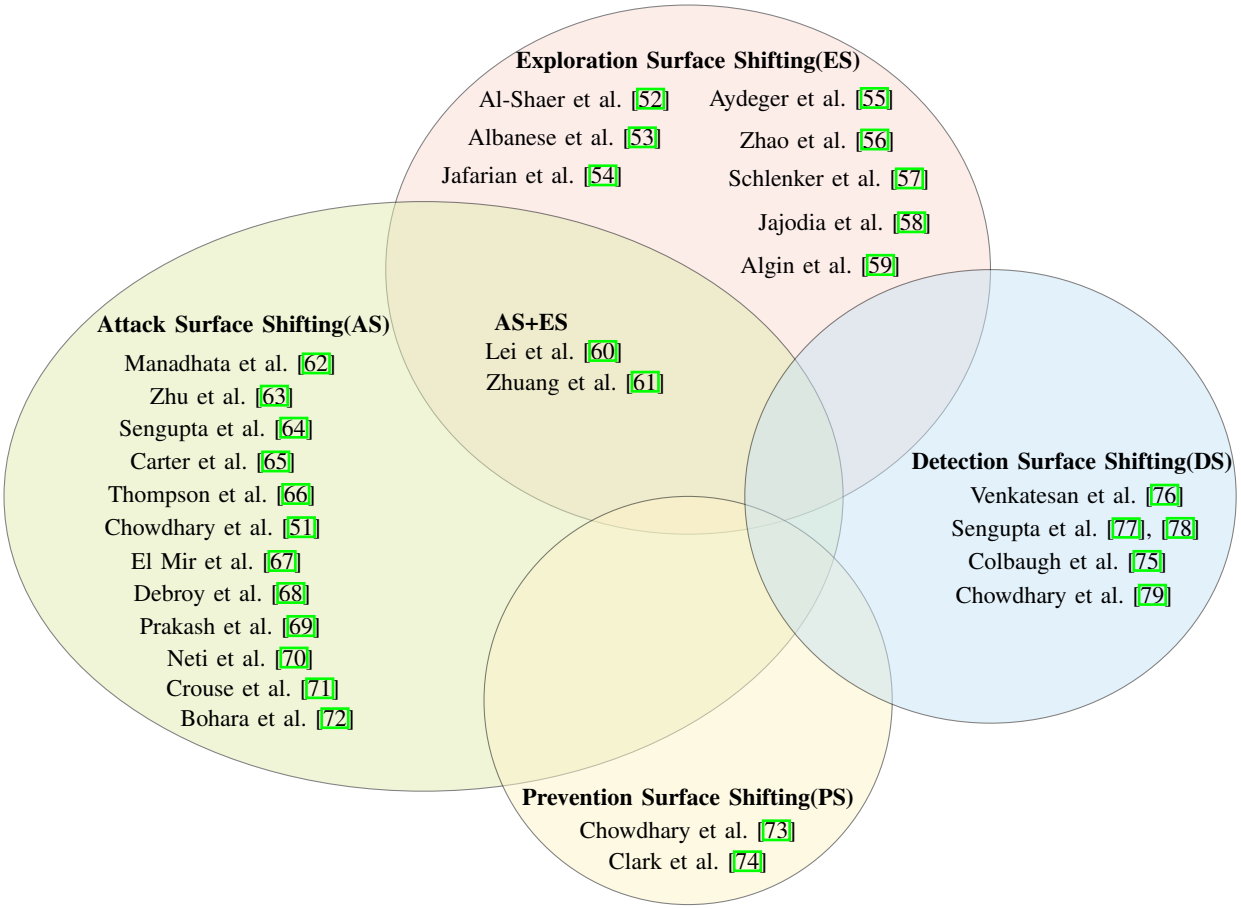


Fig. 4: The different surfaces that can be moved by a particular Moving Target Defense (MTD). Moving the exploration surface makes it harder for the attacker to figure out the exact system configuration by making the sensing actions noisy or erroneous. Moving the attack surface makes a particular attack inapplicable. Moving the detection surface, similar to 'patrolling methods', helps in providing efficient detection in budget-constrained cyber environments. Moving the prevention surface makes it harder for an attacker to ex-filtrate data even when they have a strong foot-hold inside the system.

to minimize the similarity of the attack surface in consecutive configurations and at the same time, have a minimum impact on performance. As we will soon see, this multi-objective trade-off is a common theme in many of the works on MTD.

In [63], Zhu and Bashar break down a full-stack system into several layers (denoted by l). For each layer, they have a set of technologies that can be used to provide the functionality that the layer is responsible for. Now, when all layers (or stacks) come together to form the full-stack system, all possible combinations of technologies cannot be used to provide proper functionality to the end-user. Thus, from all these possible combinations, they handpick a few combinations of technologies that meet the use-case demands for the full-stack software among which they switch, which defines the configuration space C. On similar lines, Sengupta et al. [64] also assume a full-stack web-application and thus, has similar action sets where the technologies for each layer are relevant to web-application development. The two papers differ in the way they decide *how to switch* (to be discussed later).

Carter et al. design an MTD system that switches between different Operating Systems (OSs) (which are called 'platforms') [65]. In their case, the configuration set C is the set of all OSs that they can shift between. They mention a notion of similarity (or diversity) between two configurations $\in C$ based on code syntax and semantics of the functionality provided. On similar lines, authors in [66] move away from MTD systems that consider a set of abstract configurations and implement an MTD that can perform the OS rotation for machines deployed in the systems hosted on a network using a centralized mechanism. We now shift our attention to MTDs that move elements solely relating to networks.

In [51], Chowdhary et al. describe an MTD that leverages port hopping to thwart known attacks on a cloud-based system. In their work, the states of the system are composed of variables, each of which indicates if a certain vulnerability in the system has been exploited (or not) and based on it, decides when and how to move. This fits well with our earlier discussion on how various surfaces are inter-related – the attack surface shifting comes into play when an attacker can successfully evade the detection surface. Along similar lines, authors in [67] move a deployed Virtual Machine (VM) to a different physical server if the impact of known vulnerabilities (measured using heuristic measures) on the physical server exceeds a certain threshold. In [68], the authors implement

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

MTD at a different layer of the system abstraction where they move services deployed on a particular VM to another VM. A natural extension could be to use both the layers for designing a hybrid MTD that shifts (1) software deployed on a VM and (2) individual VMs deployed on actual physical servers in the cloud network. This is similar to the concept of multi-layer MTD, as discussed in [64], [63].

In [69], authors talk about a range of configurations (precisely, 72 of them) and analyze the effect of using various game-theoretic strategies for switching between them. They shed some light on the amount of security gain that can be achieved in various settings. In [70], the authors point out a fundamental assumption that is often inherent in cybersecurity in general and MTD systems in particular– the different configurations of the MTD are not vulnerable to the same attack (similar to the notion of diversity described in [65]). They create a bipartite graph that maps hosts to vulnerabilities and show that MTD is more effective when the diversity of the configuration $\in C$ is higher. Similar conclusions have also been drawn in regards to an ensemble of Deep Neural Networks (DNN) where higher *differential immunity* to attacks leads to higher gains in robustness [80].

In rare cases, the number of configurations in C may become large. To address scalability challenges, a solution could be to first reduce the configuration set C by partitioning them into disjoint subsets [78] or choose a reasonably-sized subset and perform MTD on them separately. In [71], the authors take the latter route and use *genetic algorithm* search methods where the fitness function measures the diversity of the subset. In [72], constraining the movement of software between containers both reduces the size of the set C and helps to minimize performance impact.

*3) **Detection Surface Shifting**:* The concept of detecting attacks by inspecting traffic on the wire or the behavior of a request on a host machine has been the cornerstone of cybersecurity. Unfortunately, placing all possible Intrusion Detection Systems (IDS) on a system, especially in the case of a large network system, can lead to a degradation in the performance of the system. Thus, to minimize the impact on performance while keeping the attacker guessing about whether their attack will be detected or not, researchers have looked at intelligent ways in which a limited number of detection systems can be placed on the underlying network. These methods are similar to patrolling methods that try to identify security breaches in physical security domains like airports, wildlife sanctuaries, *etc.* with limited resources [81].

In [76] and [77], authors show that when faced with stealthy botnets and external adversaries, who are powerful enough to attack any internal node of a deployed system, shifting the detection surface helps to maintain system performance, while being effective in detecting attacks. In both of these cases, they have a set of nodes N that are deployed in the network. The configuration set C is composed of all k-sized subsets of N (where $k(< |N|)\}$). In [79], [78], the authors argue that in many real-world multi-layered networks, the assumption that an attacker can attack from any place in the network is too strong and split the configuration set C into disjoint sets based their position in on an attack graph.

In contrast to MTD for detection surface shifting, whose goal is to maximize security while adhering to performance constraints of the underlying network, some works investigate detection surface shifting for the sole purpose of enhancing security. In [75], the authors use an ensemble of classifiers that can distinguish a mail as spam and switch between them to make it more difficult for the attacker to fool the system. In general, the use of machine learning for anomaly detection [82] coupled with the paradigm of MTD for stochastic ensembles [83], [80] can lead to interesting future research, especially the introduction of a new attack surface (authors in [84] highlight several challenges in the use of supervised machine learning in cloud network settings). In these cases, each classifier is a configuration in C.

*4) **Prevention Surface Shifting**:* The goal of an MTD in shifting the Prevention Surface is to make the attacker uncertain about the defense mechanism that is in place; this forces the attacker to spend more resources and come up with sophisticated methods to exfiltrate the data. It also adds a layer of reasoning on the adversaries part; for example, distinguishing between whether their attack was undetected and went through to the actual system, or was detected and their behavior is presently being monitored becomes difficult.

Investigation of MTD techniques for shifting the prevention surface has been scarce, especially in the context of computer networks. We think this is mostly because an administrator can only use these defenses when they can identify an attack with high accuracy, which is often too strong an assumption. Yet, in some cases, authors make this assumption– in [73], an MTD mechanism that modifies the bandwidth of the network in response to malicious activity is proposed. The configuration space C consists of infinite actions as the bandwidth can take any real value. Similarly, researchers have also considered shifting the latency period when replying to a query that is, with high probability, malicious [74]. The latter work also considers the deployment of decoy nodes and switches among them to hide from an adversary seeking to actively uncover the prevention surface, i.e. the decoy nodes.

*5) **Multi-Surface Shifting**:* By choosing the shift multiple technologies that belong to different surfaces of a system, one can design an MTD that moves more than one surface. Yet, research on hybrid MTDs has been rare. In this regard, we discuss two works that try to shift both the exploration and the attack surface of a system.

In [76], authors show that constructing a simple proxy-based switching approach is ineffective by introducing a new attack known as the proxy harvesting attack that collects a set of IPs over time and then, performs a DDoS attack against all of them. To protect against such attacks, they propose an MTD approach that replaces entire proxies followed by the task of remapping the clients to these new proxies. This renders the exploration effort of the attacker useless and at the same time, reduced the attack efficiency of the attacker. As opposed to looking at a particular network-motivated setting, authors in [61] and [60] formalize the concept of MTD in which they highlight that the set $C = ES \times AS$, i.e. the elements represents configuration tuples that have technologies from both the attack and the exploration surface.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

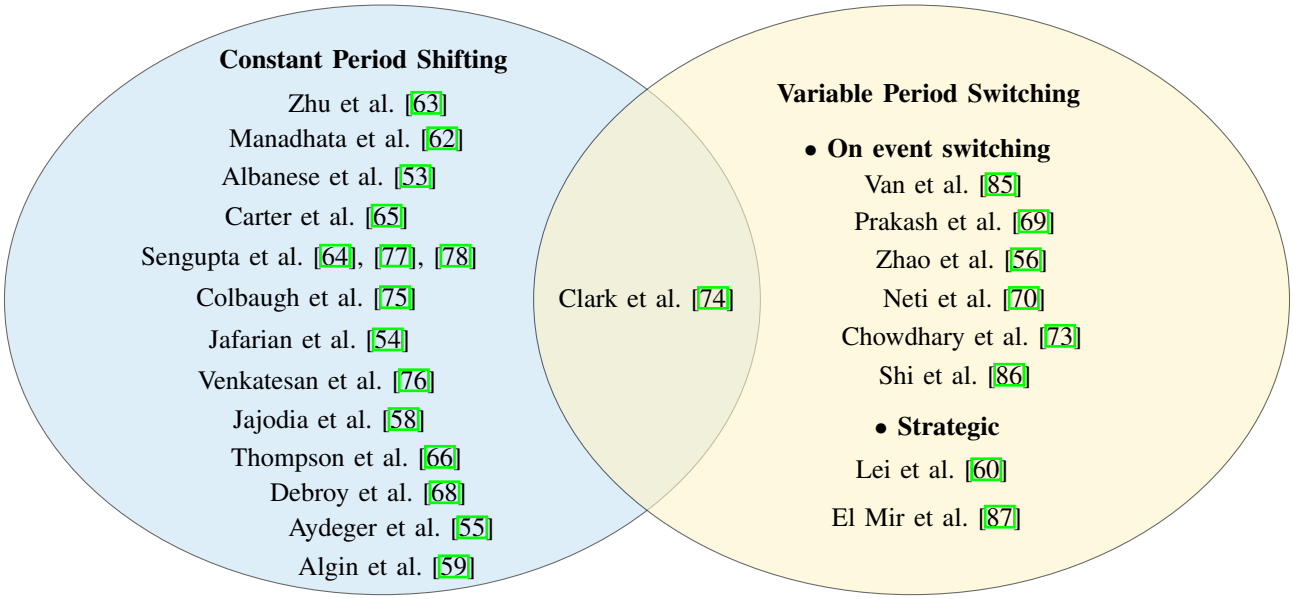


Fig. 5: The time period between two move events is either a fixed interval or decided based on some form of reasoning in the various Moving Target Defenses proposed in the literature.

An interesting future direction could be to leverage existing security research on amalgamation of multiple surfaces and extend it to setting where MTD can be used. For example, NICE combines the techniques involving the detection of attacks to countermeasure (or prevention surface) selection in a single framework [88]. This work can be the first step to propose MTD solutions that move both the detection and prevention surface.

We believe the implementation of systems that integrate multiple surface shifting mechanisms under a single MTD mechanism has several challenges. For example, some of the surfaces might lend themselves easy to management by a centralized controller like SDN, while others are better suited for movement in a decentralized fashion. Characterizing these challenges that deter researchers from considering multi-surface shifting and suggesting methods to overcome that is an interesting research direction.

B. The Timing Function T– When to switch?

Having defined the possible configurations that a defender can switch between, the next question to ask is *at what point in time does a defender choose to execute a switch action that changes their present configuration c to another configuration c'*? To answer this question, we divide the works on MTD systems into two broad categories based on how the time-period between multiple switches is determined- *Constant Period Switching* (CPS) and *Variable Period Switching* (VPS). We first describe the primary characteristics of these categories and then categorize the different MTDs (see Fig 5).

In CPS, the timing function enforces an MTD system to move, in a randomized fashion, after a constant time-period. This time is dependent on the composition of the set C because attacks on different compositions have different investments in regards to time. On the contrary, in VPS, the timing period between one switch action can vary from the time of another switch. The works in this category can be further subdivided into two distinct categories based on whether the timing function T is reflexive or strategic based on opponent modeling. While the two classes have the exact opposite definition, there is no restriction to not use both in certain scenarios. For example, when multiple surfaces being shifted, it is reasonable to have one surface shift using CPS while the other is shifted via VPS. We conclude his section with a brief discussion in this regard and later, elaborate on it in Section VI.

*1) **Constant Period Switching (CPS)**:* As mentioned, the key idea of these methods is to move the MTD system after a constant amount of time. In existing works, this is stated as an implicit design choice (for example, in frameworks that model the MTD as a game, the actions of the defender are played after a constant time-period unless they explicitly discretize the time and consider it in the state information), or an explicit one (for example, to ensure security, we move the MTD system after every 100 seconds).

A lot of the research works in MTD literature such as [89], [64], [77], [62], [58], do not explicitly bring up the topic of a time-period but inherently assume that the system moves after a constant time. In these works, authors (1) model the problem as a single-step game and generalize the solution of this game to multiple stages (not necessarily repeated games), and (2) showcase the problem of *when to switch* as a different one than *how to switch* and only address the latter, forcing the system admin (or defender) to consider a constant time-period when implementing MTD solutions.

Some research works explicitly state that the time-period is uniform and the equilibrium strategy is played at the start of each stage [63], [65], [59]. Note that, some of these works, such as [59], [53], experiment different timing functions for switching, i.e. see the effectiveness of an MTD if the constant

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

time-period was set to different predefined values. This helps them empirically figure out the constant timing function that works best. Some other works that employ a constant timing function, use different terminology. For example, authors in [76] address DDoS attacks use the notion of a pre-defined fixed frequency of switching while research in [54] refers to T as a constant mutation interval. Other works that have syntactic generalizability (by using notations such as t_i that allow a user to use different time intervals for configuration i) default to a constant timing function.

An interesting question is *how long should this constant time-period be in practice*? In [66], the authors vary the time-period from $60-300$ seconds and evaluate the effectiveness of OS rotation in a network-system based on (1) the likelihood of thwarting a successful exploit, (2) the magnitude of impact for an exploit, and (3) the availability of applications. They show that a smaller $t = 60s$ was often good enough to thwart Network Mapping (Nmap) [90] attacks, but may allow accurate OS fingerprinting when $t = 60s$. The latter does not help the attacker because, during the execution of the attack, the fingerprinted OS might simply have shifted. For $t = 300s$, these results look less promising against automated attack systems , but work well when evaluated against human experts. To allow for faster rotations, the authors set up machines with different OS and after every time interval, simply redirect traffic to a new OS. While this makes faster switching less painful, it incurs added cost to maintain multiple machines. To reduce this, we believe that having at least two systems is necessary to allow for such small switching periods, especially when shifting between OSs. With two systems, one VMs sets up an environment while the other handles traffic and the role switches in the next time step.

In [68], researchers use the knowledge obtained from historical attack data to obtain a cyber attack inter-arrival (CAIA) rate. Then, they setup an optimization problem to maximize the time-period of switching, constrained upon the fact that the constant timing function has a value less than CAIA. Along similar lines, authors in [55] have looked at *traceroute* [91] data between possible source-destination pairs to decide on a reasonable time-period for obfuscating links or mutating routes of ICMP packets on the network. This helps to mitigate crossfire attacks.

An interesting case arises in [52] because the states of this MTD represent a bipartite mapping between hosts and virtual IPs (vIP) and the authors let each edge in the mapping have a separate, but constant, mutation time. Some hosts belong to a set of High-Frequency Mutation (HFM) set while others belong to a Low-Frequency Mutation (LFM) set. To ensure that availability of a host is not impacted, the hosts that are in the HFM set map to two vIPs (one that it was mapped to in the previous round and one to which it is mapped in the present round) for a small time duration compared to the time-period of switching. In [53], the authors also have a local timer maintained by each host; its expiry triggers the change in their vIP. They use *ad hoc* messaging to communicate this change to other nodes in the network to facilitate routing.

*2) **Variable Period Switching (VPS)**:* The idea behind this switching strategy, as evident from its name, is to have a way to come up with a timing function that varies the switching time based on the present condition of the system. For example, if the system transitions through a series of configurations $\langle \ldots, c, c' \ldots \rangle$ and the time spent in c and c' is denoted by t and t' respectively, then $t \neq t'$. We believe that doing a VPS along with an MTD mechanism is similar to doing a two-layer MTD where the first layer deals with a meta MTD for shifting the time-surface and then, the second layer MTD is responsible for executing the actual cyber-surface switching. We will now categorize the MTD works under the following sub-classes.

a) On-event Switching: Most works that have a general timing function, which isn't constant, fall under this sub-category. The main idea, evident from the nomenclature, is that when a particular event occurs, such as detection of an attack, unavailability of a link or a server, the timing function specifies a time-period to switch. In most cases, the switch action is triggered immediately.

A well-known case study is that of the *FlipIt* games [85]. The true state of the system is represented using a boolean variable $1/0$ that indicates whether the defender or the attacker has control of an underlying software system. The value is unknown to both the players, and a defender strategy is based on its belief that the true value is 0 (which represents the attacker having access to the server). Thus, the timing function is dependent on this belief. On similar lines, in [56], authors update the belief about the sender type (are they malicious?) upon detection of suspicious packets on the network. This belief, in turn, influences the timing function T. Authors in [70] have a belief variable indicating whether a vulnerability was exploited or not. The timing function orders a switch action if the belief value exceeds a certain threshold.

Other works argue that obtaining belief parameters (such as *with what probability is the sender an attacker vs. a normal user?* or *with what probability is the system compromised?*) that are accurate is difficult in cybersecurity settings because they are based on analysis of historical datasets that are both scarce and might represent a different distribution. These works seek to use the intuitive knowledge of security experts in designing the timing function. In [73], if the admin detects a spike in bandwidth usage of a particular link or sub-net, they change the maximum bandwidth value allocated to that link or the network. Authors in [86] scan open connections routinely and upon detection of unexpected connections, move between MTD configurations to protect the system against port-scanning attacks.

b) Strategic: To understand these methods, we design a timing function that produced integer values in the range $[0, t_{\max}]$. These represent discrete time intervals and a value t represents the time at which a move occurs.

In [60], the authors model the MTD setting as a game-theoretic setting. While we discuss this in much more detail in the upcoming section on the design of the movement function M, we briefly highlight it here. The basic idea involves representing a game-state that considers the currently deployed configuration c and an integral-valued time variable t. In each state that can be represented by the tuple (c, t), it recommends between the actions *switch* or *stay*. The stay action takes them

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

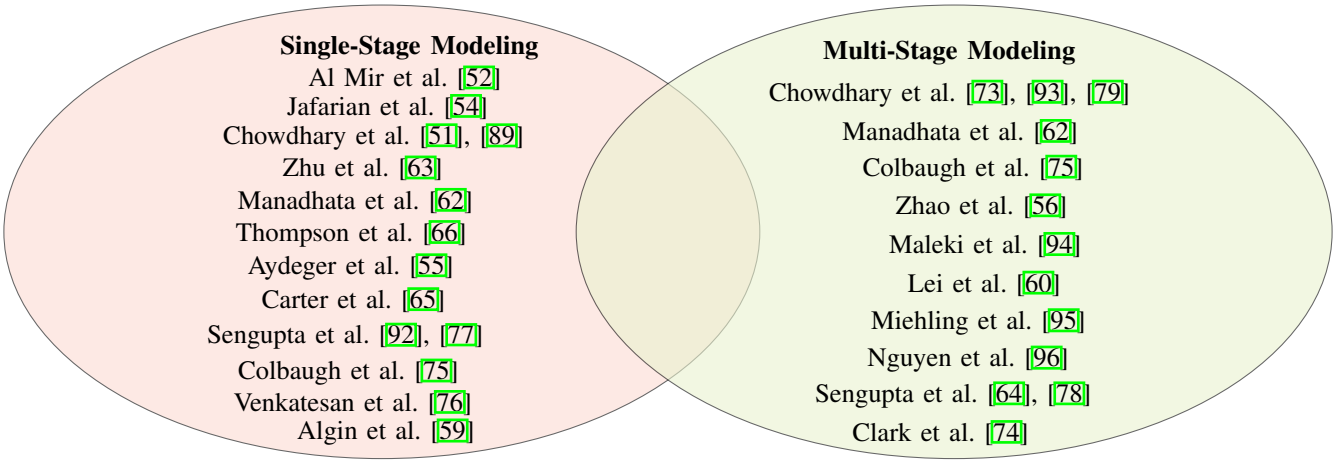


Fig. 6: Game-theoretic modeling of Moving Target Defenses (MTDs), that is necessary for obtaining the movement strategy M, can be categorized into either single-stage or multi-stage game modeling. This choice reflects the type of threat model being considered and the characteristic of the particular system for which the MTD is implemented.

to the state $(c, t+1)$, while the switch actions move them to the state $(c', 0)$. Note that the stochastic switching or movement policy described later may, with some non-zero probability predict, $c' = c$.

In general, the idea of incorporating the time variable as a part of the state can result in an explosion in the number of states and thus, computing a policy for all states time intensive. Authors in [87] show that they can model the problem similarly but resort to a simple strategy for defining the timing function. While it may be sub-optimal, scalability is no longer a concern. The timing function considers the impact of (known) vulnerabilities in the currently deployed configuration and based on it, recommends a switching time $t \in \{0, 1, \ldots, \infty\}$. If there are no vulnerabilities whose impact is greater than a predefined threshold, then the system remains static until a new vulnerability is discovered.

c) Hybrid Switching: Although the idea of having both a CPS and VPS in an MTD mechanism seems counter-intuitive at first, authors in [74] look at MTD mechanisms where one layer shifts the time window of replying to a strategic adversary using an event-based timing function while the other layer that moves the decoy nodes to hide from an active adversary is done using a constant timing function. This acts as a stepping stone for an investigation on the effectiveness of CPS *vs.* VPS for a particular surface and finally when a combination of both should be considered.

C. The Movement Function M – How to switch?

In this section, we look at how different MTDs come up with a policy for movement. While some surveys (eg. [5]) have been solely devoted to analyzing how this question, we propose a unifying framework that maps any MTD system to a Markov Game. This gives us a better sense of the implicit assumptions made by the various MTDs when coming up with M. Before getting into the categorization shown in Figure 6, we first briefly introduce Markov games [97].

An MTD system can be mapped to a Markov game defined by the tuple $\langle S, A^{\mathcal{D}}, A^{\mathcal{A}}, T, R^{\mathcal{D}}, R^{\mathcal{A}} \rangle$ where S denotes the state of the MTD system (often represented by deployed configuration c), $A^{\mathcal{D}}$ denotes the set of pure strategies that map to the set of configuration set C and the mixed strategy over this set maps to the movement strategy M. The other variables are specific to the domain for which the MTD system is designed and help in coming up with good M; $A^{\mathcal{A}}$ represents the set of attack actions, T represents the transition from one configuration to another depending on the joint action taken by the defender and the attacker, and R^i represents the rewards for the player i (i.e. the defender $\mathcal{D}$ or the attacker $\mathcal{A}$). We will use the superscripts $\mathcal{D}$ and $\mathcal{A}$ to represent the functions for the defender and the attacker respectively.

*1) **Single-Stage Modeling**:* In this section, we investigate works that have modeled the interaction between an attacker and a defender as a single-stage game. In these settings, the goal is to come up with a single mixed-strategy that can be played in all configurations (or states). Thus, the movement function $M : H \rightarrow C$ is independent of any history or $H = \emptyset$. As we shall see, the different works consider various notions of equilibrium to come up with this mixed-strategy.

In [62], the configuration of the system is modified by a pair of actions, one by the defender and one by the attacker. The defender's action, in this case, maps a value of system features represented by the set F to one of the possible actions in the set $\{enabled,\ disabled,\ modified,\ unchanged\}$. The attacker's action set $A^{\mathcal{A}}$ is comprised of conceptually opposite actions that can *dis-enable* a port, *disable* a functionality, *etc.* and the rewards, $R^{\mathcal{A}}$ and $R^{\mathcal{D}}$ for the attacker and the defender, are given by weighing the change in system's features ΔF, i.e. the change in the attack surface ΔAS and the attack surface measurement ΔASM, as follows.

$$R^{\mathcal{D}}(s, a^{\mathcal{D}}, a^{\mathcal{A}}) = B_1(\Delta F) + B_2(\Delta AS) - D_1(\Delta ASM)$$

$$R^{\mathcal{A}}(s, a^{\mathcal{D}}, a^{\mathcal{A}}) = B_3(\Delta ASM) - D_2(\Delta AS)$$

where the B_i and D_j are weights defined by security experts of the system. Note that the *cost of a defense* action (D_1) and the cost of attack action (D_2) are incorporated in the reward. Thus,

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

solving for an equilibrium of this game results in strategies that account for the cost-reward trade-off. The authors use the notion of a Subgame Perfect Equilibrium (SPE).[1]

In [52] and [54], the defender played a predefined strategy that is either uniform random (choose k random vIPs at random) or based on intuitive heuristics (select top-k scanned vIPs). The latter is based on the assumption that the most scanned vIPs will not be scanned as frequently in the new state. The chosen vIPs are then assigned to the k hosts using a purely random matching function.

In [51], [89], the authors explain the game modeling based on a real-world example and compare the effectiveness of a reactive switching strategy to the Uniform Random Strategy (URS). URS, which is found to be more effective, picks all pure strategies $a \in A^{\mathcal{D}}$ with equal probability —$1/A^{\mathcal{D}}|$. An array of works that simply use URS as the defender's policy are [66], [55], [59]. This makes an implicit assumption that the game is a constant sum and the normal form game matrix has symmetric rewards. When no information is available about the attacks and the defender has no preference over the MTD configurations, this may be a reasonable assumption. Fortunately, defenders often have an idea about an attacker's threat model and thus, such assumptions (resulting in URS) result in highly sub-optimal movement strategies. Works such as [66] closely mirror URS strategies but make a minute modification– given the present deployed OS configuration c, they consider a URS over remaining OS configurations, i.e. $C \setminus \{c\}$ as the movement function M. In [55], given all networking paths from source to destination, one is picked at random while in [59], the authors use to Fisher-Yates shuffling to select a transmission schedule at the start of every round.

In [65], [77], [92], [76], [63], and [99] the authors design a single-stage normal form game. Thus, if the defender chooses to deploy a particular configuration $c \in A^{\mathcal{D}}$ and the attacker chooses a particular exploit $e \in A^{\mathcal{A}}$, then the rewards for the players can be read from the normal-form game matrix. At equilibrium, a mixed strategy over the defender's actions turns out to be the optimal movement policy. Thus, M is a stochastic function based on the mixed strategy, chooses the next configuration at the end of each switching period. In [99], the authors consider an MTD for the Internet of Things (IoT) and show that Zero-Determinant strategy results in a dominant strategy for the single-stage game. In [63], the authors assume (1) a zero-sum reward structure, and (2) a threat model in which the attacker is irrational. The opponent model for the attacker is determined based on historical traces of the attacker's response to defense strategies and eventually updates the utilities of the game to reflect the rationality of the attacker. After each update, the resulting Nash equilibrium (NE) is played. In [65], the authors assume that if the present system is working with the operating system c and the system is made to switch to the operating system c', lower the similarity between c and c', more secure the move action. To model this, the defender's rewards are defined w.r.t. switch actions and, over time, fine-tuned by simulating the behavior of an adversary. Similar to [63], the NE strategy is chosen to be the defender's policy.

The other works assume that an attacker will eventually, with reconnaissance on its side, figure out the mixed strategy of the defender (using maximum likelihood estimates). Thus, authors of [92], [77] concentrate on finding the Strong Stackelberg equilibrium (SSE) strategy for the defender. The rewards for their game follow a general-sum structure and are obtained using the CVSS metrics of known attacks that can successfully exploit one of the defender's actions, i.e. the MTD configurations that can be deployed. On similar lines, authors in [76] use the notion of SE to obtain defender's strategy to thwart DDoS attacks.

*2) **Multi-Stage Modeling**:* Works in these sections reason about (1) the history, i.e. paths that the system has taken, which may be a result of the actions taken by the players, to reach the present state, (2) the future, i.e. how the decision in the present state affects the rewards in the future or both.

In [73], similar to the work by [62] in single-stage modeling, the authors discretize the continuous action space of the defender, which represents bandwidth values, to two levels– high and low. Then they find a meaningful defender strategy over them by ensuring that in a repeated game setting, the advantage that an attacker gained by packet flooding attacks (at some stage in the past) is neutralized by reducing their bandwidth to the low state for $1 \leq x < \infty$ number of game stages. On similar lines, authors in [75] consider a repeated game setting and analyze the defender's policy against self-learning attackers. They infer that in their case, the optimal policy converges to the URS. Other works such as [56] also update their belief about an attacker based on observations drawn from multiple plays of the game. This belief is then used to come up with an optimal strategy for the defender.

A set of works in MTD leverage the attack graph of an underlying system to better reason about the sequential process of an attack. In [95], the authors model the problem of network intrusion via a Bayesian Attack Graph. The action set for the attacker $A^{\mathcal{A}}$ includes the (probabilistic) paths an attacker can take in this graph. Then, the authors map these paths with the defender's imperfect sensing capabilities to form a Partially Observable Markov Decision Process (POMDP) [100]. Thus, the state and transition of this POMDP are designed to model the attacker's behavior and the optimal policy becomes the defender's strategy in a particular belief state. A shortcoming of this work is that this strategy may be sub-optimal if the *attacker* deviates away (intentionally or not) from the assumptions that informs the POMDP modeling. On the other hand, modeling the scenario as a Partially Observable Stochastic Game (POSG) [101] results in scalability concerns for any real-world system.

Authors in [94], [79], [102], and [60] relax the assumption about partial observability and formalize MTD in a Markov Game framework. In [94], authors consider policies over the defender's action set that comprises of either single or multiple IP hops. Each action results in the defender uniformly selecting the next state given that an attacker samples actions randomly from $A^{\mathcal{A}}$. They can show that multi-element IP hopping actions increase the defender's reward by a greater

[1] In sequential games, a strategy profile is an SPE if it is a Nash equilibrium in every subgame of the original game [98].

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

magnitude compared to static defense strategies. As we will discuss later, the authors do not model the cost of performing a hop action or the downtime associated with switches in the defender's reward function. Thus, the optimal defender policy might be sub-optimal for performance in real-world multi-layered network attack scenarios.

In contrast, the works [79], [78], [60] incorporate this trade-off in the rewards ($R^{\mathcal{D}}$ and $R^{\mathcal{A}}$) of the Markov Game, and can thus generate policies for the defender that trade-off security corresponding to an attacker's actions and performance at each step of the game. In [60], the inclusion of time variables and attack strategies in the state allows it to formulate the problem as an MDP but, becomes vulnerable when an attacker is aware of this modeling. It also makes it less scalable for large networks. In all these works, the impact on performance $C^{\mathcal{D}}$ is a part of $R^{\mathcal{D}}$ and often, just a *heuristic idea* as opposed to simulation in a real system that informs these values. The more accurate these measures become, the better is the strategy. Authors in [64] model the performance and security concerns in a different way than the above methods. The performance cost represents the switching cost while the rewards of the state represent the security costs of deploying a particular configuration. Then, they formulate an optimization problem that produces a defender strategy that reduces the cost of switching over two steps (one switch) and maximizes the security over a single step. Although the authors in [64] do not discuss how sub-optimal their switching strategies are, recent work by [103] extends their approach to a Markov Game setting with the Stackelberg Equilibrium concept.

Lastly, authors in [74] look at an MTD defense mechanism for deception (obfuscation of the links to send attacker to decoy nodes) and model attackers who are actively trying to uncover this deception over a repeated stage interaction. For one problem, they use the Nash equilibrium (NE) while for the other (identifying decoy nodes) they consider the Stackelberg equilibrium (SE) as the defender's strategy. We believe that it is necessary to highlight a shortcoming of the different modeling choices, especially in light of multi-stage attacks and APT scenarios, however optimizing the model and ensuring the scalability is a difficult task, thus we highlight these as possible research opportunities in the Section VI.

IV. Implementation of Moving Target Defenses

In this section, we discuss how the various Moving Target Defenses (MTDs) have been implemented using research test-beds and industrial products. First, we briefly discuss the tools for MTD implementation. In this regard, we highlight that traditional networking technologies like middleboxes have a set of disadvantages. To overcome this, users can leverage the advancement in networking technologies such as Software Defined Networking (SDN) and Network Function Virtualization (NFV). We briefly emphasize their role in making MTD a pragmatic solution for real-world systems. Second, we highlight how the existing MTDs, discussed in the survey as shown in table V, have been evaluated. We select a subset of works and conduct a detailed case-study to highlight how movement strategies can be implemented in practice. Third, the sub-section IV-D provides details of the test-beds used for academic research and industry products.

Middlebox	Misconfiguration	Overload	Physical/Electric
Firewall	67.3%	16.3%	16.3%
Proxies	63.2%	15.7%	21.1%
IDS	54.5%	11.4%	34%

TABLE IV: Common causes of middlebox failures. Misconfiguration is a dominant cause of failure because of different underlay and overlay network.

A. Middleboxes for enabling Moving Target Defenses

Middleboxes are the devices used by network operators to perform network functions along the packets data-path from source to destination, e.g., Web Proxy, Firewall, IDS. This provides a decentralized framework that can be used alongside existing network technology to implement strategies for MTDs. Furthermore, researchers have focused significant effort on several issues associated with middleboxes, such as ease of use, ease of management and deployment, the design of general-purpose middleboxes for different network functions *etc.* This makes them seem like a good choice for enabling the practical implementation of MTDs.

Unfortunately, a survey of various middlebox deployments conducted by Sherry *et al.* [104] reveals some drawbacks such as increased operating costs caused by misconfiguration and overloads that affect their normal functioning. As shown in the Table IV, based on the results in the survey [104] of 57 enterprise network administrators from *NANOG* network operators group, the misconfigured and overloaded middleboxes are the major reasons for middlebox failure. Furthermore, about 9% of administrators reported about six and ten hours per week dealing with middlebox failures. Also, the adoption of new and secured middlebox technologies is slow in the industry based on the survey results. In the median case, enterprises update their middleboxes every four years. The use of traditional networks for incorporation of MTD defense can increase chances of network misconfiguraion and outages.

Moreover, this static nature of the middleboxes themselves, in contrast to the dynamic nature of the system they enable, provides an assymetric advantage to the attackers as noted by Zhuang *et al.* [61]. The attackers can perform necessary network reconnaissance, identify the services and configuration of the applications, and operating systems by leveraging the middleboxes. This information helps the attacker in choosing best-fit attack methods against the static configuration and select the best time to attack the system. The attackers can use the compromised resource to maintain the foothold in the network for a long period of time and try to exploit other high-value targets in the same network.

B. SDN and NFV for enabling Moving Target Defenses

Software Defined Networking (SDN) [4] is defined as a networking paradigm that decouples the control plane from the data plane, allowing a global view of the network, and centralized control capabilities. SDN deals with packet headers, from layers 2-4 of OSI model and other protocols such

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

Research Work	SDN/NFV	Implementation Layer	Technologies/ Testbed	Maturity Level
[79], 2019	✓	network and application	Sample Use-Case	simulation
[105], 2018	✓	network and application	ONOS [106]	emulation
[57], 2018		network	Personalized use cases	simulation
[58], 2018		abstract	Personalized use cases	simulation
[93], 2018	✓	network and application	Personalized use-cases	simulation
[89], 2018	✓	network	Personalized test-bed with VMs	emulation
[107], 2018		application	Polyverse	commercial
[33], 2018		network and application	Personalized Test-bed with VMs	simulation
[96], 2018		network	personalized use-cases	simulation
[108], 2018		network and application	Deception Grid [108], Crypto Trap [109]	commercial
[77], 2018		network and application	Mininet	both
[110], 2017	✓	network	Opendaylight Helium [111]	simulation
[112], 2017	✓	network	Ryu SDN [113]	simulation
[114], 2017		application	LLVM compiler [115]	emulation
[64], 2017		application	personalized use-cases	simulation
[116], 2017		application	CMS application [117]	simulation
[118], 2017		application	Simpy [119]	simulation
[73], 2017	✓	network	network simulator + ODL controller	emulation
[60], 2017		abstract	Personalized Test-bed with VMs	simulation
[56], 2017	✓	application	Mininet with POX controller	emulation
[86], 2017	✓	network	CloudLab [120]	emulation
[121], 2016		network	Smart grid, energy management system (EMS)	emulation
[64], 2017		application	personalized test-bed	simulation
[122], 2017		network and application	Morphisec	commercial
[59], 2017		network	Personalized Simulation Environment	simulation
[55], 2016	✓	network	Mininet with Floodlight controller	emulation
[94], 2016		network	Personalized use-cases	simulation
[51], 2016	✓	abstract	OpenStack [123] with ODL [111] controller	emulation
[124], 2016		network	OpenStack [123]	emulation
[68], 2016	✓	network	GENI [125]	emulation
[76], 2016		network	Rocketfuel Dataset	simulation
[126], 2016		network	ARCSYNE [127] & SDNA [128], [129]	emulation
[87], 2016	✓	network and application	Mininet with POX controller	simulation
[130], 2016	✓	network	CyberVAN	commercial
[68], 2016	✓	application	GENI [125]	emulation
[69], 2015		abstract	Personalized simulation environment	simulation
[131], 2015		network and application	Cyber Quantification Framework	emulation
[74], 2015		network	matlab	simulation
[95], 2015		network	personalized use-cases	simulation
[132], 2015		network and application	RPAH framework	simulation
[66], 2014		network and application	CORE Impact Pro	commercial
[65], 2014		application	personalized test-bed	simulation
[133], 2014	✓	network	Personalized Cloud System	emulation
[134], 2013		network and application	PlanetLab [135]	emulation
[88], 2013		network and application	Personalized virtual cloud system	emulation
[85], 2013		abstract	Personalized test-bed	simulation
[63], 2013		application	game-based theoretical framework	analytic
[54], 2012	✓	network	NOX in Mininet	emulation
[70] , 2012		abstract	Proposes to use a modified CTF environment	analytic
[71] , 2012		network and application	Personalized test-bed	emulation
[75], 2012		network	KDD dataset [136]	simulation
[52], 2012	✓	network	Personalized use cases	simulation
[137], 2011		network and application	SLAAC [138]	simulation
[139], 2011		network	MUTE	simulation

TABLE V: A taxonomy of MTD Implementation categorized defenses based on the use of SDN/NFV, the layer of network protocol stack they protect, key technologies they use, and the level of maturity at which they have been implemented.

as MPLS [140]. SDN and NFV have emerged as a state-of-the-art network architecture for data centers and backbone networks. Google's *B4 project* [141] shows the feasibility of SDN for handling real-world network challenges such as traffic engineering and Quality of Service (QoS).

The decoupling allows a network architecture where switches act as forwarding devices, maintaining flow tables populated with flow rules. The new architecture allows considerably more flexible and effective network management solutions, and a unified programmable interface that can be utilized by software developers for application deployment [142]. The SDN architecture can be vertically split into three layers described below:

- **Application Plane:** It consists of end-user business applications that consume SDN communication and network services. The examples include network security or virtualization applications.
- **Control Plane:** The control plane consists of SDN controllers such as ONOS [106], OpenDaylight [111] providing open Application program interfaces (APIs) to

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

monitor network forwarding behavior. The communication interface between the application and control plane is known as northbound interface. The interface between control and data plane is known as southbound interface.

- **Data Plane:** The forwarding elements such as OpenFlow switches are present in the data plane. The forwarding devices can be physical or virtual switches. Control plane installs flow rules in order to govern the forwarding behavior of data plane devices.

Network Functions Virtualization (NFV) [3] has emerged as a technology that provides a virtualized implementation of hardware-based equipment such as firewall, routers, and Intrusion Detection Systems (IDS). Virtual Network Functions (VNFs) can be realized through virtual machines (VMs) or containers running on top of the physical server of cloud computing infrastructure. SDN acts as an enabling technology for NFV and help in centralized management, making it easier to implement MTD. Despite the great benefits offered by SDN and NFV, its use to implement MTDs has been limited to a few research works (as can be seen in Table V).

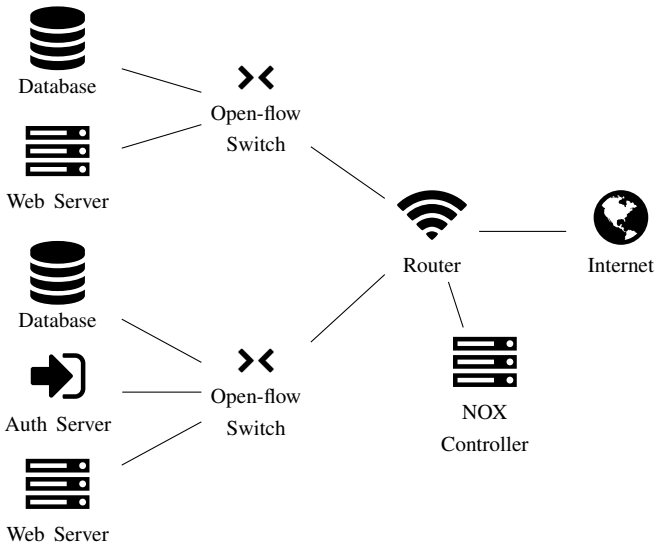


Fig. 7: OpenFlow-based Random Host Mutation mutates the virtual IPs of hosts based on a pool of available IP addresses.

C. SDN-based MTD Applications and Case Study

We categorize the MTD industrial, and research implementations from the perspective of networking technologies used, e.g., traditional network or SDN/NFV as shown in the Table V, as highlighted in column 2. It is noteworthy that SDN/NFV has been a dominant technology for MTD research. Around 34% of the research works in Table V, utilize SDN/NFV for implementation of MTD or cyber-deception. Column 3 describes the layer in network protocol stack where the describe MTD solution has been implemented. As on may notice, most research works are implemented at the network or the application layer. In the fourth column of the table, we identify the key technologies used by these research works; if they utilize a particular implementation test-bed (for example, GENI [125], then the name of the test-bed is mentioned in this column. Column 5 of the table shows the level of *maturity* of the MTD research work. We categorized the research works into four levels - *analytic* - if only numerical results or thought-based experiments have been presented in the said paper. We put the research work under the category *simulation* - if the research work describes some simulated network, e.g., *Mininet*. The *emulation* category consists research works that use close to real world networks, e.g. multiple VMs deployed in a cloud testbed, e.g., GENI [125]. Lastly, if the research work has been deployed in some commercial product dealing with live attacks in a production grade network, we consider these research works under category *commercial*.

We observed that more than 50% of the research works use simulated networks or applications when experimenting with MTD techniques, whereas $\sim$ 34% of research works have used emulated environments for performing MTD based analysis research. Only few MTD solutions e.g., $\sim$ 13% have implemented MTD at commercial level (in production grade networks for dealing with live attacks). This shows that though MTD has been well accepted in research community, and benefits of MTD can help in dealing with different threat models, the industry adoption of MTD is rather slow. The key reason behind this is the adverse impact that MTD can induce on Quality of Service (QoS), and the additional cost-overhead associated with deployment of MTD. We discuss these factors under *Qualitative* and *Quantitative* metrics in Section V.

1) SDN-based Network Mapping and Reconnaissance Protection: The first step of the Cyber Kill Chain (CKC) is the identification of vulnerable software and OS versions. Most scanning tools make use of ICMP, TCP or UDP scans to identify the connectivity and reachability of the potential targets. The replies to the scans can also reveal the firewall configuration details, i.e., what traffic is allowed or denied. The time to live (TTL) information can also help in the identification of a number of hops to the attack target [133].

SDN-enabled devices can help in delaying the network attack propagation by hiding the real response and replying back with a random response in order to confuse the attacker. As a result, the attacker will believe that random ports are open in the target environment. The attack cost will be increased since the attacker will need to distinguish the real reply from the fake reply. SDN-enabled devices can also introduce random delays in TCP handshake request that will disrupt the reconnaissance process utilized by the attacker for identification of TCP services. The cost-benefit analysis of MTD adaptations against network mapping attempts has been discussed by Kampankis *et al.* [133]. The survey considers cost-benefit aspects of MTD in Section V, under the quantitative metrics of MTD.

2) Service Version and OS Hiding: The attacker needs to identify the version of OS or vulnerable service in order to mount an attack. For instance, the attacker can send *HTTP GET* request to Apache Web Server, and the response can help in identification of vulnerability associated with a particular version of the Apache software. If the attacker gets a reply 404 Not Found, he can identify some obfuscation happening at the target software. A careful attacker can thus change the attack vector to exploit the vulnerability at the target.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

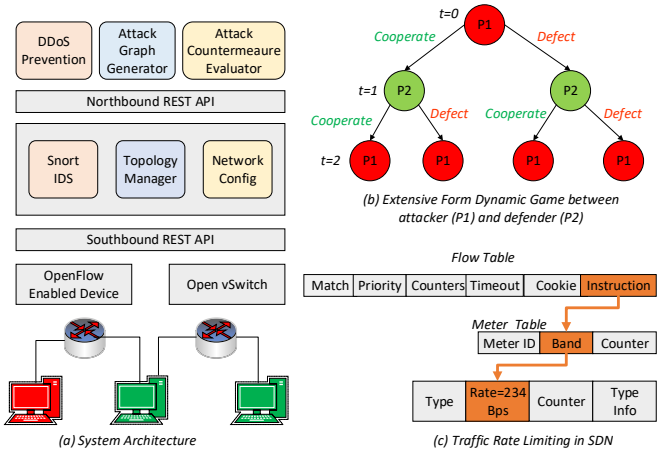


Fig. 8: SDN/NFV based MTD that models the interaction between an attacker and the defender as a game to select an optimal countermeasure strategy. The work imposes limiting on bandwidth and views the MTD as an adaptation problem.

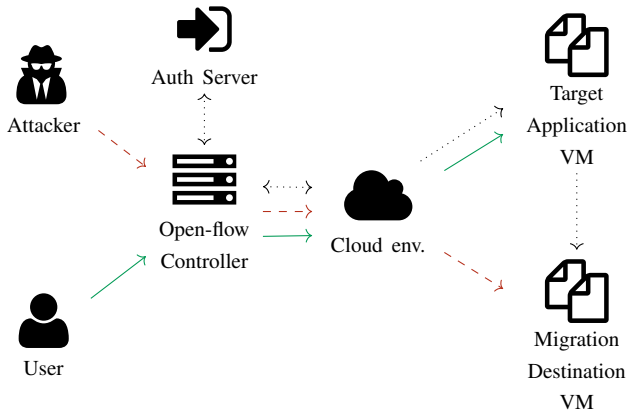


Fig. 9: Virtual Machine migration using frequency minimal MTD seeks to migrate applications to new VMs and redirect an attacker's DDoS attacks without effecting availability to regular users. The goal of this work was to find an optimal timing function T, termed as the migration frequency.

An SDN-enabled solution can override the actual service version with a bogus version of the web server. Some application proxies leverage this technique to prevent service discovery attempts by a scanning tool. Another attack method is known as Operating System (OS) Fingerprinting, where the attacker tries to discover the version of the OS which is vulnerable. Although modern OS can generate a random response to TCP and UDP requests, the way in which TCP sequence numbers are generated can help an attacker in the identification of OS version.

In an SDN-enabled solution, the OS version can be obfuscated by the generation of a random response to the probes from a scanning tool. SDN can introduce a layer of true randomization for the transit traffic to the target. The SDN controller can manage a list of OS profiles and send a reply resembling TCP sequence of a bogus OS, thus misguiding the attacker.

3) Protection against multi-stage attacks and service disruption: Once the attackers have obtained necessary information, they proceed towards targeted attacks, with the aim of stealing information - SQL Injection or service disruption - Distributed Denial of Service (DDoS). The SDN-based MTD can introduce various countermeasures at network-level such as network shuffling [110], route modification [105], IP, and port obfuscation as discussed by Wang *et al.* [86].

We now discuss some case studies which show use of SDN/NFV capabilities for implementation of MTD techniques - what, when, and how to switch? Jafarian *et al.* [54] use OpenFlow based random host mutation technique to switch virtual IP (what to switch) targeted by reconnaisance attempts. Debroy *et al.* [68] use SDN framework to identify optimal rate of migration (when to switch) and ideal migration location for VMs under DoS attack. Chowdhary *et al.* [73] use SDN-environment to create a analyze the hosts mounting DDoS attacks on critical network services. The SDN-controller downgrades network bandwidth (how to switch) using a Nash-Folk theorem based punishment mechanism.

► **Case Study (What to Switch)– OpenFlow Random Host Mutation.** SDN makes use of OpenFlow protocol for control plane traffic. Jafarian *et al.* [54] propose OpenFlow enabled MTD architecture as shown in Figure 7, can be used to mutate IP address with a high degree of unpredictability while keeping a stable network configuration and minimal operational overhead.

The mutated IP address is transparent to the end host. The actual IP address of the host called real IP (rIP), which is kept unchanged, but it is linked with a short-lived virtual IP address (vIP) at regular interval. The vIP is translated before the host. The translation of rIP-vIP happens at the gateway of the network, and a centralized SDN controller performs the mutation across the network. A Constraint Satisfaction Problem (CSP) is formulated in order to maintain mutation rate and unpredictability constraints. The CSP is solved using Satisfiability Modulo Theories (SMT) solver.

Sensitive hosts have a higher mutation rate compared to the regular hosts in this scheme. The OF-RHM is implemented using Mininet network simulator and NOX SDN controller. OF-RHM is able to thwart about 99% of information gathering and external scanning attempts. The framework is also highly effective against worms and zero-day exploits.

► **Case Study (How to Switch)– Dynamic Game-based Security in SDN-enabled Cloud Networks**. DDoS attacks are a major security threat affecting networks. Attacker's leverage sophisticated bots to generate a huge volume of network traffic, and overwhelm critical network services as discussed by Chowdhary *et al.* [73]. The case study presented in this research work, focused on the MTD decision *how to move?* The framework presented in Figure 8 (a) to communicate with OpenFlow devices using southbound REST API, whereas any application plane decision and network analytic is performed using northbound REST API.

The Snort IDS [143] detects malicious DDoS patterns, and uses Nash Folk Theorem [144] to analyze the behavior of a malicious node, e.g., red color VM shown in Figure 8 (a). The

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

scenario is represented as an extensive-form game spanning multiple rounds. If the node sending network traffic P2, the Defect - Figure 8 (b), the normal bandwidth B is reduced to $\frac{B}{2}$. The bandwidth is reduced in each subsequent round by network admin P1, till average bandwidth over a period of time $t = \{0, 1, 2, ..\} \in T$ is B.

The SDN controller utilizes Instruction field present inside the flow table in order to change the band rate of the node flagged as malicious, as shown in Figure 8 (c). Thus SDN-based rate-limiting serves as an effective countermeasure against flooding/loss of availability attacks like DDoS.

► **Case Study (When to Switch): Frequency Minimal MTD using SDN.** *Frequency minimal MTD* [68] approach considers resource availability, QoS and exploits probability for performing MTD in an SDN-enabled cloud infrastructure. The design goals of this research work are as follows:

1) Identification of optimal frequency of VM migration, ensuring minimal wastage of cloud resources.
2) Finding the preferred location of VM migration without impacting the application performance.

As shown in Figure 9, the normal clients can access the services hosted in the cloud network via regular path, and the attack path represents the path along which the attacker tries to exploit the target application. The VMs periodically share their resource information such as storage and compute capacity with the controller along the control path. Depending upon the level of threat, the migration of VM can be proactive or reactive. The real IP address of the VM hosting cloud application is hidden from the clients. The data-path shows the path along which VM application and its back-end database are migrated. VM migration is based on the following factors:

- **VM Capacity:** This parameter considers the capacity of migration target in terms of computing resources available to store files and database of current and future clients.
- **Network Bandwidth:** The lower the network bandwidth between the source and target VM, the slower will be the migration process and the longer will be the exposure period (in case of active attacks) for the VM being migrated. This parameter considers bandwidth between source and target while performing VM migration.
- **VM Reputation:** This is the objective indicator of VM robustness to deter future cyber attacks. It is the history of VM in terms of cyber attacks launched against the VM. This parameter is considered in order to ensure VMs suitability for migration.

This research work estimates the optimal migration frequency and ideal migration location based on the parameters described above. The VM migration mechanism is highly effective in dealing with denial-of-service (DoS) attacks.

D. MTD Testbeds: Research and Prototyping

The practicality of MTD can be established by the deployment of MTD techniques and tactics over an underlying network. In this section, we identify some platforms which can assist MTD researchers in conducting experiments or serve as a guideline when creating MTD case studies.

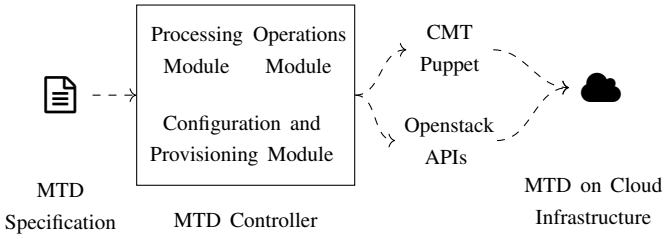


Fig. 10: A platform that takes an abstract specification of a cloud system as input and outputs the corresponding system on a cloud infrastructure. Advantages of cloud automation using Automated Enterprise Network Compiler (ANCOR) include performing live instance migration.

1) GENI Platform: Global Environment for Networking Innovation (GENI) [125] provides a distributed virtual environment for at-scale networking and security experimentation. Individual experiments can be conducted in isolation using network sliceability (ability to support virtualization and network isolation). Each experimental slice is provided with network and computing resources. The users can request resources for an allotted time period, in which they can conduct experiments and release the resources once they have finished the experiment.

Debroy *et al.* [68] utilize GENI framework for implementation of a VM migration MTD solution. The authors utilized the InstaGENI platform at the Illinois site to host a news feed application targeted by DoS attacks. The setup also involved four non-malicious users, one remote SDN controller, and attacking VMs, all hosted at physically distributed locations. The attacker VMs were utilized for sending a large number of HTTP GET requests to news feed site in order to achieve a DoS attack. Proactive and reactive frequency minimal (FM) MTD countermeasures were deployed in response to targeted DoS attacks. The research work shows the capability of GENI platform to host similar MTD experiments where users can study the impact on network bandwidth, VM performance, attack success rate when MTD security countermeasures are implemented at scale on a cloud platform.

2) OpenStack Cloud: OpenStack [123] is a cloud operating system that consists of compute, storage, and networking resources. The users can log in through GUI to provision isolated virtual networks or utilize OpenStack APIs to create and configure the network. OpenStack is compatible with the existing virtual solutions such as VMWare ESX, Microsoft's Hyper-V, KVM, LXC, QEMU, UML, Xen, and XenServer.

Mayflies [124] utilized OpenStack for designing a fault-tolerant MTD system. The research work utilizes VM introspection to checkpoint the current state of the live node/VM, and reincarnation - node commission/decommission based on attacks against the introspected node. The strategy allows the Mayflies framework to deal with attacks in a short interval of time, avoiding the attack progress. Zhuang *et al.* [145] conduct MTD experiments to simulate a pure-random MTD strategy and an intelligent MTD approach based on intelligent attack indicators. The experiments were conducted using *NeSSi2* [146], an open-source, discrete event-based network

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

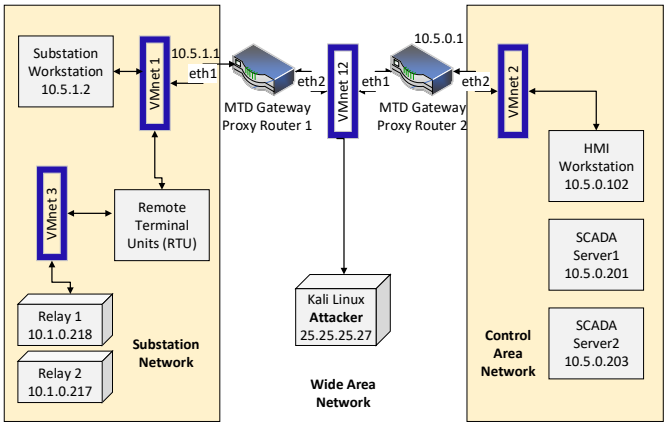


Fig. 11: An MTD-based architectural platform for IP hopping. The solution has been targeted at Supervisory Control and Data Access Systems (SCADA). The MTD gateway routers act as dynamic proxies, periodically mutating the IP address of the external interfaces exposed on the public wide-area network.

security simulator. The authors proposed implementation on an OpenStack based MTD testbed as future work.

MTD CBITS MTD-platform for *Cloud-Based IT Systems* (MTD CBITS) [147] as shown in Figure 10 evaluates the practicality and performs detailed analysis of security benefits when performing MTD on a cloud system such as OpenStack.

The platform makes automated changes to the IT systems running on the cloud infrastructure. One adaptation performed in MTD CBITS is replacing the running components of the system with new ones. The system parameters are stored in the operational model and can be viewed using the MTD system inventory - CMT (Puppet) APIs. Any MTD adaptation is also recorded in an operational model for future reference. OpenStack API utilizes Puppet agents to communicate with the live instances of the cloud system. MTD controller communicates with agents over a private network.

3) CyberVAN Testbed: [130] provides a testing and experimentation platform for cybersecurity research. The platform supports high fidelity network, by representing the network in a discrete event simulator. CyberVAN allows hosts represented by VMs to communicate over the simulated network. The testbed utilizes network simulator *ns-3* [148] to simulate network effects such as end-to-end network latency, link capacities, routing protocols, *etc.* The platform also supports wireless networks by modeling mobility, interference, propagation effects, as well as the details of different waveforms. The Scenario Management GUI allows the experimenter to access and manage the elements of an attack scenario, including network traffic and logging, running analytic tools on the VM, saving results of the experiment, pausing and restarting the experiments, *etc.*

4) MTD SCADA Testbed for Smart Grid Network: In the context of smart-grid environments, Pappa *et. al.* has proposed an MTD for IP hopping [121] (see Figure 11). The network consists of two SCADA servers running Siemens Spectrum Power TG Energy Management Systems (EMS) software. The software periodically polls Remote Terminal Units (RTUs) for network traffic measurements. The substation networks are connected to the Wireless Area Network (WAN) using MTD gateway routers. The remote attacker can utilize the publicly exposed service over the WAN to identify security vulnerabilities. The services can be exploited using traditional means or using APT techniques. The gateway routes in the network act as dynamic proxies, mutating IP address of the external interfaces, while providing end-to-end SCADA communication. The IP generation algorithm employs a random initial seed to initiate the IP generation at the network boot up. Both routers are assigned set (j, k) of n random IP addresses at the start, in combination of initial seed vector. The initial seed vector ensures order between the two gateway routers is synchronized; this helps to prevent IP collision and network outages. Additionally, the MTD algorithm used creates a dynamic network technology which makes the job of network mapping difficult for the attacker.

5) MTD Commercial Domain Solutions: **TrapX:** The production-grade decoy network technologies such as *DeceptionGrid* [108] and *CryptoTrap* [109] deceive the attackers by imitating the true assets. DeceptionGrid provides agentless visibility and control appliance, that dynamically identifies and evaluates network endpoints and applications when they connect to the main network. Assuta Medical Center incorporated DeceptionGrid as a part of their network security suite. The use of this framework helps to counter not only known attacks but also APTs and zero-day attack scenarios. The DeceptionGrid created a network of traps (decoys) that camouflaged real medical devices such as MRI & CT scanners, X-ray machines, and ultrasound equipment (PACs). The solution has been deployed on many VLANs across Assuta Medical Center and provided better visibility into the lateral movement of the attackers.

CryptoTrap on the other hand is utilized for countering the ransomware early in their exploitation lifecycle. This helps in countering malware propagation while protecting valuable network assets. The traps (decoys) are masked in the form of SMB network shares across the network. The fake data of the company is replicated across the traps. Once the attacker touches these traps, the targeted computer is disconnected from the network and security administrator is alerted about the incident. In effect, the attacker is held captive in the trap and attacker actions are logged for further threat intelligence.

Polyverse: *Zero-day* vulnerabilities ,e.g., *Heartbleed* [149] (vulnerability discovered in *OpenSSL* software in 2014), can cause a significant damage in an underlying network. There is no known attack signature to identify these vulnerabilities, hence they can bypass security monitoring software unnoticed. The attackers trying to target software or OS memory-based zero-day vulnerabilities start with the assumption that the gadgets they are trying to access are located at a certain address or within a specific offset from the absolute base address. *Polyverse* [107] employs MTD strategy to defeat this assumption of the attackers. The polymorphic version of Linux is utilized by Polyverse in order to create a high-level of entropy in the software system in such a way that the entire memory structure is diversified. With the polymorphic versions of Linux, the entire Linux software stack is roughly

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

randomized. The resulting program is semantically identical to the original program (functionally equivalent), however, nearly every machine instruction is changed.

MorphiSec: Some threat vectors classified as Advanced Evasive Attacks cloak the malicious intent in order to deceive the security monitoring tools. Some of these techniques include Polymorphism (changing malware signature), Metamorphism (changing malware code on the fly), Obfuscation (hiding malicious activity), behavior changes (waiting for normal user activity before executing). *Morphisec* [122] uses MTD at network-level (route changes, random addresses and ports), firewall level (policy changes), host-level (OS version, memory structure changes), and application level (randomizing storage format addresses, application migration, multilingual code generation) in order to deceive and detect attacks using evasive attack techniques.

V. Evaluating the Effectiveness of Moving Target Defenses

In this section, we present a list of qualitative and quantitative metrics that can help in determining the effectiveness of MTD. First, we discuss the quality of the defender actions C, i.e. the various system configurations it can choose to deploy, in terms of performance and security. This discussion helps in identifying a set of qualitative metrics. As we will see, most works either consider this metric implicitly when choosing the configuration set or simply ignore it. Second, we devote a sub-section on qualitative evaluation metrics. These are mathematical functions to help a defender capture measures such as the cost-benefit, the risk analysis, *etc.* of an MTD. As we shall see, they can be measured by simulation on test-beds or by using metrics based on security domain knowledge such as CVSS, attack graphs representations *etc.*

A. Qualitative Evaluation

When using a Moving Target Defense (MTD) that moves between multiple system configurations, we would want to believe that it increases the security of the deployed system without negatively impacting the performance for legitimate users. Thus, we look at how different MTDs consider these aspects either in the modeling of the defense or its evaluation. In particular, we look at two major aspects– security considerations and performance considerations– to evaluate the quality of an MTD. Under each of these sub-headings, we consider the quality of each individual defense and then, the overall ensemble of constituent defenses. This helps us identify non-trivial heuristics to understand when an MTD might succeed or fail. For example, an MTD that lacks *differential immunity*, i.e. all the constituent system configurations ($\in C$) are all vulnerable to the same attack, can never be secure regardless how well it is implemented in practice.

In regards to quality metrics, we categorize the various MTDs in Figure 12. An MTD is part of a set if they explicitly consider the particular qualitative metric when designing either the configuration set C (i.e. *what to move?*), the timing function T (i.e. *when to move?*), or the movement function M (i.e. *how to move?*). Note that only a subset of works discussed under security considerations (i.e. shown in the blue and/or yellow boxes on the left of Fig. 12) are featured under performance considerations (i.e. shown in the pink and/or green boxes on the right of Fig. 12). We discovered that majority of the defenses proposed only consider (and therefore highlight) the improvement with regards to security and ignore the impact on performance. We will now discuss the details about how the different works capture the various qualitative aspects that we put forth in this survey.

1) Security Considerations: We first look at works that reason about the security risks of individual actions followed by works that reason about the security risks associated with an ensemble and lastly, discuss works that consider both.

a) Considers only Security of Individual Defenses: Most MTDs that consider some form of game-theoretic modeling, consider the security of individual defenses in the ensemble by representing them as a part of the defender's utility value [60], [77], [58], [92], [65], [77], [151], [79]. For most of these works, the security metrics considered for each action are obtained using scoring metrics designed by experts such as the Common Vulnerability Scoring Services (CVSS) discussed earlier. These works are able to come up with an intelligent movement strategy M based on reasoning over known vulnerabilities (more specifically, common vulnerabilities and exposures (CVE)) for which CVSS scores are readily available. This can result in highly sub-optimal behavior when faced with zero-day vulnerabilities. In that regard, authors in [65] model the security of a configuration as inversely proportional to the probability with which an adversary can come up with a new attack given the attacks it performed in the earlier time steps. In [151], the authors try to model zero-day attacks by asking security experts to annotate how effective they think a particular countermeasure or defense action will be against zero-day vulnerabilities. As the annotations might be inaccurate (due to the lack of actual black-hat hackers who invent zero-day attacks in the set of security experts who annotate the defense methods), the effectiveness of their MTD cannot be accurately determined. As to how, if possible, utility values can capture the security risks associated with zero-day vulnerabilities is an interesting question and remains to be explored by future works.

Other works that also only capture the security of individual constituent system configurations are more domain or problem specific in nature. In [73], each defense action, before being played, is weighed based on the penalty it imposes on an attacker (who tried to do a DDoS attack) over a repeated game setting. In [67], authors choose an action (to migrate a VM or keep running) after reasoning about the security risks associated with the present vulnerable state. In [68], the security risk is based on the reputation of the current state, which in turn looks at the type and number of attacks that were done in previous time steps when the particular system was deployed. On similar lines, researchers in [63] find a more compact way to model and continuously update the risk associated with deploying a particular defense action. Some works that consider the longitudinal effects of a movement strategy M model the epistemic effects of movement on an attacker's knowledge about the network. For example,

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

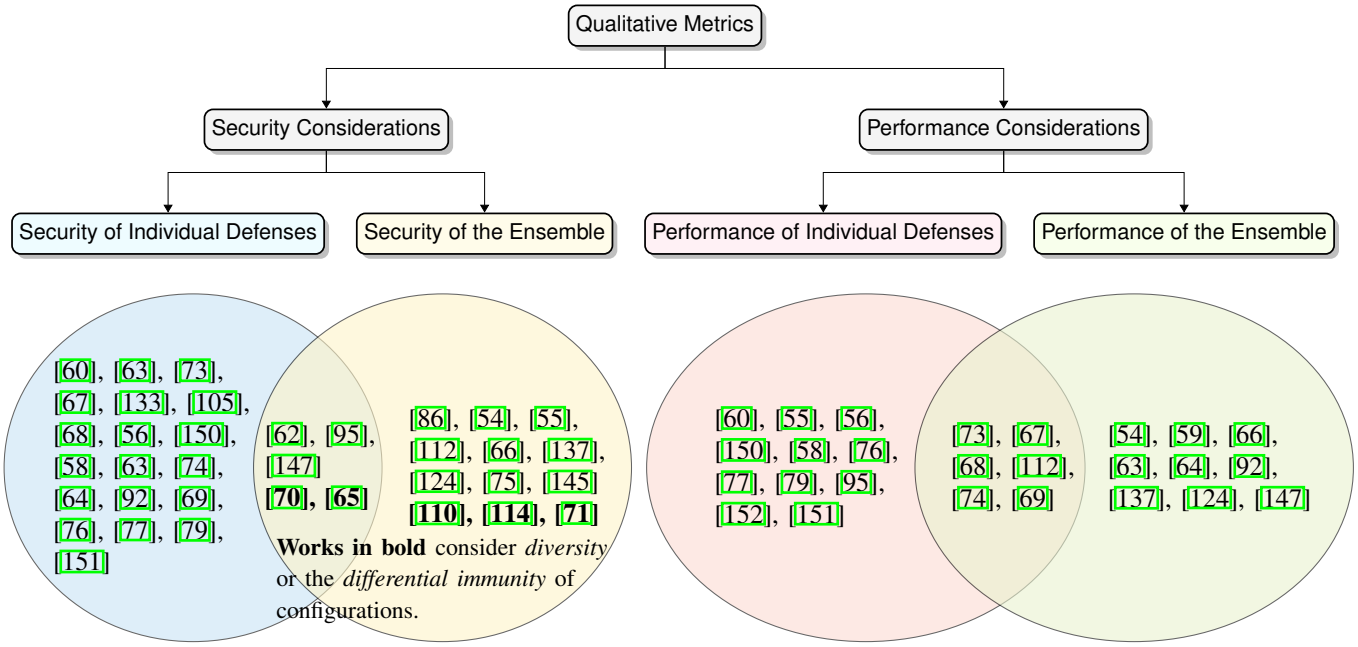


Fig. 12: A Moving Target Defense is more effective if it considers the security and performance impacts of the system configurations, both in unison and also when used as part of an ensemble that the MTD leverages. We categorize the works which explicitly consider these metrics either while modeling the defense or during its evaluation. Note that there is no work which considers all four kinds of metrics, i.e. falls in the intersection zone both on the left and on the right.

[56] considers the fingerprint of the overall network when a particular defense action is selected, while [150] reasons about the topology information a particular defense action leaks to an attacker. In contrast to all these works, [133] models the security risk associated with a particular defense action as inversely proportional to the cost (it believes) an attacker would spend to compromising it.

b) Considers only Security of the Ensemble: Works that measure only the security of an ensemble as, opposed to security of actions, mostly showcase the security benefits of the MTD ensemble by comparing them to the security benefits provided by a static system configuration [86], [54], [55], [66], [145]. In essence, these works consider the static system as the control case but unfortunately, do not ensure that this static system is the most secure constituent defense configuration in the ensemble. This way, they might overestimate and thus, over-promise the security guarantees of the designed MTD.

On the other hand, works that consider the security metrics associated with the ensemble (as a whole) at modeling time look at metrics that are similar to *entropy* or *diversity* of the ensemble. In [137], the authors try to use the large address-space of IPv6 to their advantage and are able to create more uncertainty for an attacker who is trying to pinpoint the address to use for a successful attack. On similar lines, authors in [124] argue that fast reincarnations of a functional system (i.e. bringing the service down and starting it up on a new location/container, *etc.*) increases the entropy and makes it more costly for an attacker to continuously keep attacking such a system. On a different note, researchers in [110] try to select a defense configuration from the ensemble based on how diverse it is with respect to the current configuration that might have been attacked. In order to do this, they use a topological distance measure, which in their case is the symmetric difference between the edge sets of the current and the consecutive defense configuration. Although they do not explicitly recognize it as a diversity metric like [114], [71], they bring to light an interesting issue that most MTD papers seem to either miss or assume by default. If there was an attack, that with extremely little modification, could exploit all the defender's configuration that is a part of the MTD, an MTD would not be an effective defense strategy. For example, MTD work that randomly selects a classifier for malware detection assumes by default that each classifier is not vulnerable to the same attack [75] which seems to be an incorrect assumption to make given current research works [30], [80]. We believe that it would be easier to convince practitioners about the effectiveness of an MTD if researchers can show that diverse defender actions can indeed be generated at a low cost. This is the goal of [114], who create an approach at the compiler level to increase software diversity and [71], who use a genetic algorithm to draw a pool of configurations that maximize diversity.

c) Considers both: Only a few works take a holistic view in regards to security and consider both the security of each individual defense and the ensemble as a whole. In [62], the general level at which they define the utility values, one can capture the security risks associated with an individual defense action in terms of attack surface features and attack surface measurements while the security risks associated with an ensemble can be captured via the utility variable for attack surface shifting. In [147], the security for individual defense actions is trivial for their settings, where defending a node that

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

is not a stepping stone is not beneficial at all for the defender. The security risks associated with an ensemble are evaluated through experimentation with a static defense. Particularly, the try to increase the number of interruptions for an attacker to start from one point in the network and reach a goal node. In [95], the selection of the entire ensemble is based on the fact that the set of possible defenses have a global property of covering each of the leaves that an attacker might want to reach and each individual defense action has some utility in terms of security associated with it. Lastly, the works [65] and [70] model both the diversity of constituent defenses present in the MTD ensemble and also model the security of each individual defense. In [70], each configuration is mapped to a set of vulnerabilities and thus, a diverse ensemble is composed of system configurations that do not have a lot of overlapping vulnerabilities. In [65], since they use Linux based operating systems, the diversity is modeled in terms of lines of code that is different between two Operating Systems.

2) Performance Considerations: As mentioned above, a large set of works for developing MTDs focus on showcasing the security benefits and sweep under the rug the performance costs. Note that in the case of MTDs, the impact on performance may arise due to a variety of reasons. First, each system configuration (individual defense action), that is a part of the MTD ensemble, has a performance cost associated with it and moving to a high-cost configuration impacts the performance. These concerns are termed as performance considerations for individual defense. Second, the switching from one configuration to another may need a defender to deal with (1) downtime, (2) legitimate requests on the wire that were meant for the previous configuration in a graceful manner and/or (3) keep all the different configurations running (at least two of them) to facilitate a faster switch. All these costs can be termed as shuffling costs and are categorized as performance considerations of the ensemble because they only arise when there is an MTD ensemble.

We will now describe how the various MTD systems consider the performance costs associated with each individual configuration followed by the performance cost associated with the ensemble. We also discuss works that reason about both these factors together and analyze the affect on the overall Quality of Service (QoS) when such defenses are is deployed.

a) Considers only performance of individual defense actions: Similar to the case of security metrics, when we look at performance considerations for an individual defense, most game-theoretic works model these as a part of the defender's utility function [60], [76], [77], [79], [96]. Various equilibrium concepts in these games yield movement strategies for the defender that gives priority to constituent defenses that have low performance costs while ensuring that the security is not impacted by a lot. In [60] and [96], the reward functions are defined at an abstract level and the authors point out that they can be used to consider the performance cost of constituent defenses. In [76] and [77], the authors consider the impact of placing Network-based Intrusion Detection Systems (NIDS) on the latency of the network and use centrality based measures as heuristic guidance for it.

A large number of works also look at problem-specific

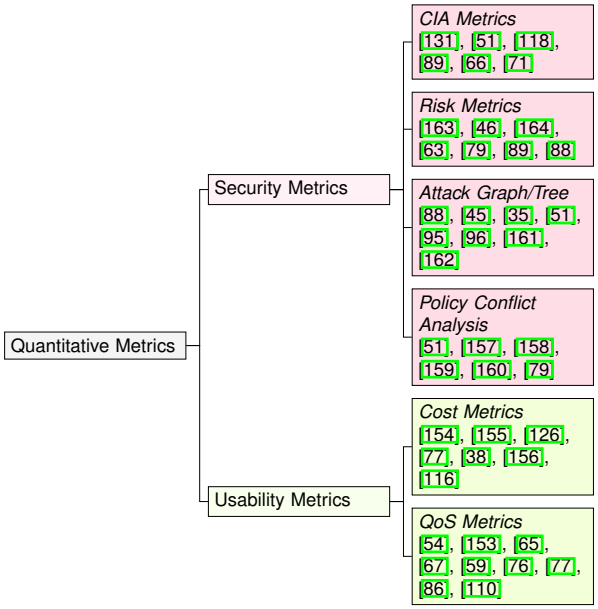


Fig. 13: Categorization of MTDs based on the fine-grained quantitative metrics, under the broader umbrella of security and usability metrics, they use for evaluation.

instances. Authors in [55] perform experiments to evaluate their MTD against crossfire attacks and find that the average time for packet transfer from one point to another increases because a particular path selected at random can be highly sub-optimal. On similar lines, in [56], the authors consider the performance cost of doing a *defend* action on legitimate user traffic. On a different note, [150] tries to solve a multi-faceted problem where the MTD tries to obfuscate the network topology to an attacker and, at the same time, ensures that it does not negatively impact a defender's ability to debug network issues. This is done by leveraging the knowledge asymmetry about the network topology that a defender and an attacker has. In [58], the performance costs of a *honeynet* configuration, which is the defense action for this MTD, represents the number and *quality of resources* (or honey) necessary for developing a credible honeynet that fools an attacker. Lastly, authors in [151] combine both the costs of setting up a good defense and the usability of that defense for legitimate traffic as the performance cost for a particular placement of countermeasures.

b) Considers only performance of the ensemble: Multiple works try to capture the performance costs that result because of the dynamics involved is shuffling between the different configuration but do not look at the *performance impacts* resulting because of a particular *bad constituent configuration*. In [63] and [64], the authors consider the *one-step cost* of switching from one defense action to another one and seek to find a strategy that minimizes this.

Other works, instead of accounting for the performance impact of the ensemble, compare an MTD defense to a static system configuration by measuring usability metrics such as latency, availability to legitimate users, *etc.* In [137], authors notice an overhead of 40 bytes for the IPv6 header and latency

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

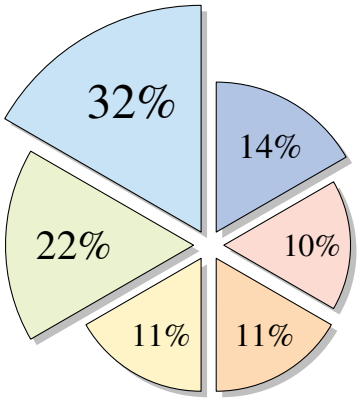
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

of 12ms during address change as opposed to 3ms when MTD is not implemented on the network packets. They point out that a more efficient implementation might help in reducing this gap. On similar lines, authors in [124] highlight that creating an entire file system replica takes two more minutes in the case of an MTD system in comparison to a non-MTD enabled system refresh. On the contrary, [147] does not notice negligible performance overhead for instance replacements in a 14 node system (where one is a controller node and the others are compute nodes). Although, they do notice a larger number of HTTP error packets on the wire when using the MTD. To ensure that the usability to legitimate users is not impacted at all, authors in [66] keep multiple systems running with the different configurations. At every switch, they simply pick the system that serves the request. On the downside, they incur the cost of maintaining multiple services (at least two). An interesting side effect of using MTD in [59], mainly done to thwart selective jamming attacks, is the reduction of delay in transmitting packets over the network. Existing approaches that schedule packet delivery in a deterministic way land up in packet collision scenarios. The random schedule selection in each round is shown to reduce the number of collisions, improving end-to-end (ETE) packet delivery time.

c) Considers both: A small section of works either consider or evaluate their MTD in regards to both performance of single constituent defenses and the ensemble. In [73], the authors model the performance costs associated with each defense action as a part of the utility values and consider the shuffle cost as a part of the rewards the attacker and a legitimate user gets in a repeated game setting. On similar lines, in the empirical analysis of MTDs in the context of Flip-it games [69], the authors model the cost of each defender action as a part of defender's utility value while the state of the system after a move action considers the number of servers being controlled by the defender– an indirect way of measuring system performance. Authors in [67] consider the availability of each defense action over a *bounded time horizon* and for *shuffling costs*, consider the downtime or unavailability that results when the system is migrating from one system configuration to another. On similar lines, the authors of [68] model the *host capacity* and *network bandwidth* associated with each system configuration and also determine the next configuration based on the performance of other defenses in the previous time steps. For some works, the performance costs associated with the ensemble do not arise due to the shuffle but represent the cost of ensuring exact same performance across defense configurations [112] or cost of implementing a system that can support the various configuration [74]. More specifically, authors in [112] create a system that ensures that the logical virtual identifier distances between any two nodes remain the same, while in [74], the authors assume an extra cost of creating an ensemble that is not necessary when not using MTD solution.

B. Quantitative Metrics

Although we realize that the quality of MTD is key is determining the quality of defense being offered, quantifiable measures about these qualitative terms are essential for effective evaluation. For example, shuffling x % number of hosts leads to y % reduction in attack success probability, and z % increase in overhead/quality of service (QoS) for the normal user. In this sub-section, we present the quantitative analysis of MTD research works discussed in the survey. We categorize quantitative analysis into *Security Metrics* and *Usability Metrics* as shown in Figure 13.



Reconnaissance	[54] [52] [118] [8] [9] [56] [154] [57] [62] [69] [71] [86] [89] [132] [139] [94] [95] [96] [133] [110] [112] [137]
Vulnerability Exploitation	[92] [64] [65] [67] [70] [66] [72] [96] [165] [116] [166] [124] [147] [167] [114]
(D)DoS Attack	[105] [55] [9] [68] [73] [132] [139] [154]
Multi-Stage Attack	[161] [79] [51] [60] [93] [166] [168] [145]
APT and Data Exfiltration	[85] [126] [33] [169] [151] [126] [76]
Other types of Attack	[74] [89] [88] [75] [77] [126] [76] [80] [53]

TABLE VI: Threat Models considered in the MTDs proposed in the surveyed papers.

*1) **Security Metrics**:* An important aspect of network defense is representation and visualization of network attacks. Enterprise networks are becoming large and complex with different network overlay and underlay technologies. The adage *what can't be measured cannot be effectively managed* applies aptly here. Security metrics such as the ones shown in the Figure 13, covers attack quantification using CVSS metrics– Confidentiality, Integrity, and Availability (CIA). This metric also considers attack representation methods (ARMs)– Attack Graphs and Attack Trees. We now discuss the MTD research works from this perspective of security metrics.

We surveyed $\sim$ 70 MTD research articles and analyzed them from the perspective of different threats models they targeted. A summary of the findings has been provided in Table VI. The key observations are as follows a) 32% of research works focus on defense against reconnaissance attempts, b) 22% of these research works target vulnerability exploitation, c) 11% research works use MTD defense against DoS/DDoS attacks. There has been a limited focus on using MTD defense for dealing with stealthy attacks like APT and data-exfiltration (7/68 $\sim$ 10% research works). Also, some papers used other types of attacks such as timing-based attacks [74], and network intrusions [89], [88], [75], [77].

CIA Metrics: *Confidentiality*, *Integrity*, and *Availability*

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

(CIA) are used as quantitative metrics for measurement of impact on system under attack. Zaffarano *et al.* [131] use confidentiality metric for measuring information exposed by modeling tasks. The mission *M* confidentiality valuation *v* is expressed as, *Conf (M, v)* = $\frac{1}{|T|}\sum_{t\in T}(t, unexposed)$. As the measurement equation suggests, the goal is to maintain information as unexposed over time ($t \in T$). Similarly *Integrity* valuation *v* for mission M is quantified as $Int(M, v) = \frac{1}{|T|}\sum_{t\in T}(t, intact)$. High integrity is ensured by keeping information intact over time.

Conell et al. in [118] consider availability as an important metric for analyzing impact of MTD countermeasure. The system reconfiguration rate α is modeled as a function of system resources using Continuous Time Markov Chain (CTMC) modeling. The analysis of the effect of reconfiguration on the availability is considered for fine-tuning MTD decision. MASON [89] framework utilizes Intrusion Detection System (IDS) alerts and vulnerability score *CVSS* calculated on the basis of CIA values to identify critical services in the network. A *Page Rank* based threat scoring mechanism is utilized for combining static and dynamic information, and prioritizing network nodes for MTD countermeasure port hopping. It is noteworthy, that port-hopping for $40 - 50\%$ services can help reduce overall threats in the network by 97%. This research work, however, does not consider the usability impact induced by the MTD countermeasure.

MORE [66] framework shows reduction in reconnaissance attempts and exploits targeting software integrity violation on frequent rotation of Operating Systems (OS) of the network under attack. Experimental analysis shows that a rotation window of 60s makes *nmap fingerprinting* attempts ineffective. Temporal and spatial diversity have been used to introduce genetic algorithm based MTD by Crouse *et al.* [71]. The experiments on average vulnerability of different configurations show decaying vulnerability rates for evolved configurations selected from the chromosome pool of VM configurations.

Attack Graph/ Attack Tree: CVSS present only a piece of quantitative information about the vulnerabilities such as complexity of performing network attack, impact on confidentiality or integrity of the system if the attack is successful, *etc.* This information alone is not sufficient for taking MTD decisions. Attack representation methods (ARMs) such as *Attack Graph* [45] and *Attack Tree* [35] answer questions such as (a) What are the possible attack paths in the system (b) What attack paths can be taken by the attacker to reach a specific target node in the network. SDN based scalable MTD solution [51] makes use of attack graph-based approach to perform the security assessment of a large scale network. Based on the security state of the cloud network, MTD countermeasures are selected.

Bayesian attack graphs have been used by Miehling *et al.* [95] for defending the network against vulnerability exploitation attempts. The defender's problem is formulated as a Partially Observable Markov Decision Process (POMDP) and optimal defense policy for selecting countermeasures is identified as a solution for the POMDP.

The security analysis of a large-scale cloud network in real time is a challenging problem [170]. Attack Graphs help in identification of possible attack scenarios that can lead to exploitation of vulnerabilities in the cloud network. The attack graphs, however, suffer from scalability issues, beyond a few hundred nodes as we discussed in Section II.

Risk Metrics: Like any other system, MTD systems have an associated risk once an organization considers deploying whole or part of the MTD technique. According to the *National Institute of Standards and Technology* (NIST), there are several attacks, service disruptions, and errors caused by human or machines that may lead to breaking benefits and critical assets at the organization or national level. Risks assessment is a critical measure and has many ways to deploy and use. In this survey, we identify and highlight research work that has adopted and took into consideration risks associated with deploying MTD solution. Specifically, we emphasize on the research work that evaluates the cost of the adopted MTD solution, since system administrators need to take into account the cost of using the MTD solution.

Risk assessment in the MTD has a direct and strong relationship with the effectiveness of the deployed MTD technique. According to [163], the system administrator can determine how good the MTD solution is by examining the associated risk. Therefore, the authors [163] studied the effect of deploying each MTD technique alone (i.e. shuffle, diversity, and redundancy) by inspecting the Hierarchical Attack Representation Method (HARM). HARM is basically a *ARM* at the upper layer such as attack graph (AG), and another ARM in the lower layer such as attack tree (AT), where these to ARM have a one-to-one mapping between them. Moreover, the authors also studied the associated risk by computing the instance measure (IM) which uses vulnerability's base score, impact score, etc [46].

Feedback-driven multi-stage MTD has been proposed by Zhu *et al.* [63] for dealing with multi-stage attacks like Stuxnet [171]. Author's quantify the damage or cost caused by an attacker at different stages of the network. The game between attacker-defender is modeled as a finite zero-sum matrix game with a bounded cost function, and a mixed-strategy Saddle Point Equilibrium (SPE). Players utilize cost-function learned online to update MTD strategies. The numerical results show that feedback mechanism allows network defense to respond to unexpected (exogenous) events and reduce unusual peak of risk imposed by different vulnerabilities.

In [126], the authors provide metrics for MTD evaluation and risk analysis. For risk metrics, they proposed statistical metrics to study the effect of how the attacker can quickly conduct and succeed in adversarial attacks. The authors assumed the system will always have a running task that can be measured. The validity of the metrics was studied by simulating the APT attack scenario, where they assumed the APT will always have some sort of overhead that can be measured and detected. Finally, the performance of the proposed metrics was measured also by examining the CPU utilization of the designed system.

Cheng *et al.* [60] consider the game theoretical formulation of MTD systems; specifically, they model it as a Markov Game. The authors provided a theorem, subject to probabilistic constraints, to calculate the revenue for the defensive and

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

offensive approaches in MTD systems. Their work depended on testing different defensive and offensive strategies and was tested using a networking setup that includes vulnerable services and a firewall component as well.

Another work considered the statistical approach to evaluate the likelihood of a successful attack is the work conducted by [76]. The authors proposed an approach to determine the minimum effort required a system to detect stealthy botnets. Moreover, the entropy was measured to determine how close an adversary is to the detection point, where high entropy indicates the attacker is far in distance from the detector. The evaluation of the proposed approach was conducted on a real ISP network obtained from [172]. The results of the proposed algorithm show that the detection strategy has a complexity of $O(N^3)$ although theoretical complexity analysis indicates the algorithm is $\sim O(N^6)$.

Chung *et al.* [88] provide a detailed and comprehensive evaluation for the optimal countermeasure selection over a set of vulnerable attack paths in the attack graph. By evaluating the CVSS score of vulnerabilities, the authors determined the countermeasure selection option, taking into account the ration of the Return of Investment (ROI). Countermeasure option that produces the smallest ROI is considered the optimal one. The system performance of *NICE* [88] is proven to be efficient in terms of network delay, CPU utilization, and the traffic load.

To study the effect of a number of intrusions on the system's threat, Chowdhary *et al.* [51] use a statistical approach also to evaluate how the number of intrusion in the system, with the number of vulnerabilities, affect the threat score. The threat scoring algorithm is similar to *Page Rank* algorithm. To evaluate the proposed work, two experiments were conducted. One experiment is to test the threat scoring engine on software vulnerabilities and IDS alerts. The second experiments study the effect of port hopping attack. The results show that as the number of services in the system increase, the service risk value remains unchanged between a specified interval. However, the first few services show an increase in the number of risk value. Finally, the port hopping attack showed a reduction in threat score.

Policy Conflict Analysis: The MTD countermeasures such as network address switching can dynamically and rapidly insert new type of traffic, or new flow rules (environment managed by SDN). Pisharody *et al.* [157], [158] show how different MTD countermeasures such as network address change, load-balancing, and intrusion detection can cause security policy violations. The research works discuss SDN-based MTD, but the policy conflicts can cause security violations [159], loops, and blackholes in the network as discussed by Khursid *et al.* [160]. The violations of network-wide in-variants and security policies must be analyzed before deployment of MTD countermeasure. This research challenge is an important quantitative metric, which has not been considered by a lot of research works. We discuss this as a possible research oppurtunity in Section VI.

*2) **Usability Metrics**:* This category as shown in the Figure 13, analyzes MTD research work from the aspects such as QoS (network bandwidth, delay), impact on existing mission metrics and the cost of deploying MTD defense.

QoS Metrics: MTD can induce some performance cost on the existing system resources. Jafarian *et al.* [54] identify the virtual IP (vIP) mutation, range allocation, and range distribution constraints in order to minimize the QoS impact which can be induced by vIP collisions, as well as, maintain optimal-level of unpredictability. Probabilistic performance analysis of MTD reconnaissance defenses has been conducted by Crouse *et al.* [153]. The research work analyzes quantifiable MTD metrics such as reconnaissance, deception performance, *attack success probability* vs *connection drop probability*, attacker's success probability under different conditions such as network-size, number of vulnerable computers.

Taylor et al. in [126] use mission and attack metrics for analyzing the effectiveness of a network defense. The research work analyzes dynamic defenses such as *Active Re-positioning in Cyberspace for Synchronized Evasion* (ARCSYNE) and *Self-shielding Dynamic Network Architecture* (SDNA) using mission and adversary activity set. Mission *Success*, i.e., the rate at which mission tasks are completed, and Mission *Productivity*, i.e., how often are mission tasks successful are used as QoS measurement metrics are used for evaluations.

A statistical analysis of static *vs.* dynamic attacks against different MTD strategies– uniform, random, diversity-based, evolution-based, and optimal– has been conducted by Carter *et al.* [65]. Experimental results on performance *vs.* adaptability shows that diversity-based MTD is the optimal strategy against most attack scenarios. They also show that uncertainty about the adversary type– slow adversary or fast-evolving adversary– can adversely impact the effectiveness of an MTD.

El-Mir [67] models performance parameters such as availability, downtime, and downtime cost using a Continuous Time Markov Chain (CTMC) model. The experimental results showed that cost-effective VM migration can be performed in an SDN-based network with limited impact on network performance. The research work utilizes normalized CVSS score as a key metric for initiating VM migration.

Sengupta *et al.* [77] analyze the performance impact of placing IDS (NIDS and HIDS) at all possible enforcement points in a cloud network. It is noteworthy that the placement of more than 15 detection agents in their simulated network fails to provide any additional intrusion detection benefit, whereas the network throughput decreases drastically from 16 Gbps in the case of a single detection agent to ~ 6 Gbps when 15 detection agents are placed.

CHAOS [86] analyzes how the delay intentionally introduced by MTD impacts the packet count in a SDN-managed network. The SDN controller utilizes host-mutation and decoy-severs to deceive an adversary. The obfuscated network increases the cost and difficulty for an adversary targeting the network. The percentage of information disclosure reduces from 90% to 10% in a CHAOS protected network, with slight impact on packet delay (1s to 1.5s for 1800 packets).

Cost Metrics: Protection against Distributed Denial of Service (DDoS) attacks is one of the important priorities for many cyber systems. Wang *et al.* [154] present a cost-effective MTD solution against DDoS and *Covert Channel* attacks. Through MTD adaptation, their work [154] aims to answer two main questions: 1) what is the adaptation

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

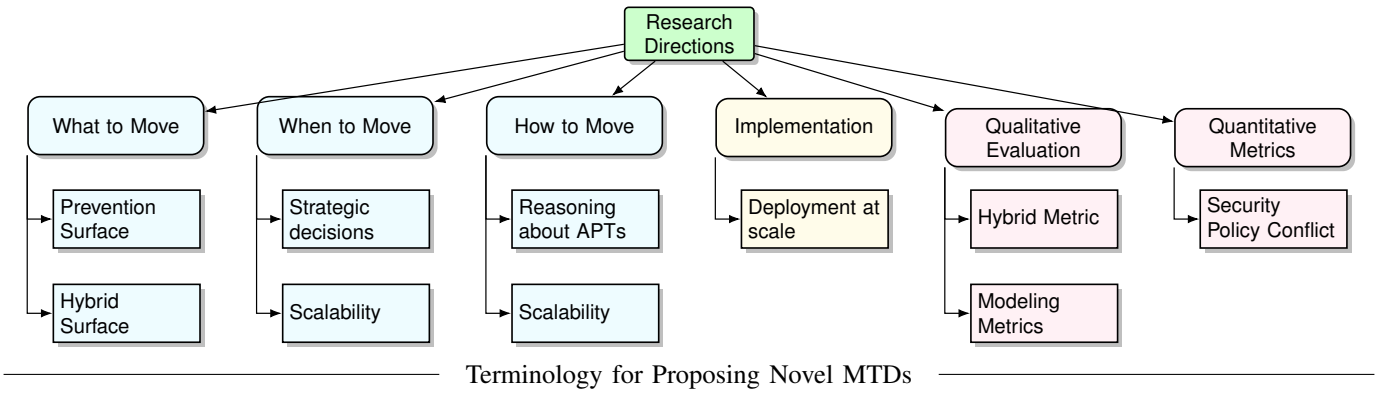


Fig. 14: Our categorizations help in identifying various directions that have been under explored during the development, implementation and evaluation of Moving Target Defenses (MTDs). In addition, it also provides a terminology that can help to underlying identify assumptions made by existing work and describe future MTDs.

cost?, and 2) what is the cost incurred by a defender if an attacker succeeds in exploiting a particular vulnerability?. This solution does not rely on IDS-generated alerts while making the adaptation. The adaptation cost includes any cost related to purchasing required software or hardware that helps in the adaptation process. Lei *et al.* [155] utilize change-point analysis method for evaluation MTD cost-benefit for a multi-layer network resource graph. The proposed method analyzes mission productivity (ΔM), and attack success productivity (ΔA) on dynamic network address translation (DNAT). The evaluation results show reduced attack success probability using DNAT over a network under observation. The path enumeration mechanism used in this research work can, however, suffers from scalability challenges because of frequent path probability calculation and update operations. The cost and effectiveness evaluation of reactive and proactive network defense strategies, has been conducted by Taylor *et al.* [126] using Measurement of Effectiveness (MOE) metrics. The research work considers hop-delay for different attack success rates, and static defense policies. They show that an attacker's productivity, i.e., how quickly attacker can perform adversarial tasks increases against static defense, whereas attacker's confidentiality, i.e., ability to remain undetected is same for both the static and the dynamic defense case.

VI. Research Opportunities

In this section, we highlight some lessons learnt and the under explored aspects of Moving Target Defenses (MTDs). These lead to a discussion on promising directions for future research (a brief summary is highlighted in Figure 14).

A. The Configuration Set (What to move?)

Works in MTD have concentrated mostly on the movement of the exploration, the detection and the attack surface. While moving the exploration and the attack surface helps a defender take away the advantage of reconnaissance that an attacker has, the goal of moving the detection surface is to improve the scalability and Quality of Service (QoS).

The movement of the prevention surface, which comprises of security modules such as Firewalls, IPS *etc.* has only been investigated by a couple of works (see discussion in Section III-A4). An MTD research can consider exploring Next-Generation Firewall (NGFW) architectures that combine security modules such as firewall, content filter, anti-virus *etc.* to provide a multi-layered defense-in-depth solution. Some current implementations of NGFW, that can be leveraged for testing the effectiveness of these defenses, are *Cisco ASA* [173] and *PAN Firewall* [174]. Further, with the rise of mobile technologies, MTDs can prove to be effective defenses. Although we discuss a few works on thwarting jamming attacks [59] and identity shifting in mobile ad-hoc networks [53], MTDs for different surfaces of in mobile infrastructure networks can be a promising direction for future research.

As previously stated, the movement of different surfaces in a single framework, although challenging, can provide greater security benefits than the movement of a single surface. Investigation in this area would require one to identify sets of configurations across the various surfaces that are compatible (in terms of performance) with one another. This prevents the number of strategies from multiplying uncontrollably by leveraging the expertise of system designers.

The categorization of the various software surfaces opens up the possibility of considering other logical surface level distinctions and thus, MTDs that shift these surfaces. For example, *Microsegmentation* [175] is a method of creating secure zones in data centers and cloud deployments to isolate workloads from one another and secure them individually. With MTD formalism, one can test the existing hypothesis and develop new ones for microsegmentation. We believe that formal modeling, in line with [79], one might discover that advanced services will be more effective when applied at a granular-level (as close to the application as possible in a distributed manner).

B. The timing function (When to Move?)

The timing problem has mostly been ignored in previous works on MTDs. While some works perform empirical studies to test the best (constant) time-period for switching, they can be highly specific to threat model and the elements being shifted. For example, while 15 second time-periods are shown

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

to be reasonable when protecting against jamming attacks [59], 60 seconds time periods are needed to defend against Network Mapping attacks [90] attacks. Also, the complexity of changing the virtual IP address vs. the underlying virtual machine impose different constraints on the lower bound on feasible time-periods.

Note that the handful of works that empirically determine time-periods are by no means complete. We need an extensive study just to come up with reasonable time periods for particular surfaces, how it effects that attack model and provide guidelines on efficient implementation methods.

On the other hand, a few existing approaches that do address the timing problem theoretically, suffer from scalability issues. In [60], the authors land up increasing number of states in their Markov Game by including time as a parameter. Inferring the defender's strategy in such Markov Games cripples the MTD to work beyond small networks. Effective solutions or improved modeling could both be interesting research directions for the future.

C. The Movement Function (How to Move?)

Randomization in the movement is a necessary part of effective MTDs. Given the extensive use of randomization in cryptography, MTD defenses tend to use a Uniform Random Strategy (URS) an the movement strategy [61]. Several works have argued that often the defender has performance information about their system and known attacks that can be used to exploit their system. In such cases, a game-theoretic modeling can result in better strategies that offer higher gains in terms of both security and performance metrics. Unfortunately, the latter works suffer from scalability issues, encouraging practitioners to default to URS. Works that can leverage existing knowledge without suffering from scalability issues can fill an important gap in MTD research.

Recently, it has been shown that attacks on networks are better characterized by persistent approaches such as APTs. In such scenarios, attackers are known to demonstrate sequential attack behavior spread over a long time. A relatively new line of work investigates modeling MTDs to come up with strategies that are effective against APTs. Although these approaches leverage the use of attack graphs to bootstrap the modeling process [169], [151], they do not scale well. While real-world cloud services host thousands of nodes, researchers have only considered small-scale settings [79], [78]. Furthermore, works such as [95] and [96] that try to consider partial observability limit the scalability even further. Study on the design of approximation approaches and their effectiveness when compared to scalable baselines such as URS will be key for future research.

Current research often makes strong assumptions about the threat model. This makes results regarding the effects of such defenses questionable. In the future, we hope to see more studies that try to figure out realistic attack scenarios. Figuring out an attacker's behavioral model might also lead the modeling community to relax assumptions often made by rationality of an attacker.

D. Qualitative and Quantitative Evaluation

As seen in Figure 12, none of the existing works demonstrate either empirically or model the impact of a proposed MTD on all four types of metrics. The lack of testing against real-world attackers also makes it difficult to prioritize which metrics a defender needs to care about and to what extent.

Beyond human studies to understand attack behavior, use of MTDs may introduce a new attack surface in cyber-systems. For example, firewall filtering rules need to be carefully analyzed to prevent conflicting security policies that may arise at the movement time because such conflicts might result in either dropping legitimate user packets or introducing new attack points. Although some works such as [51] have tried to address this issue of identifying security policy conflict for an SDN-managed cloud network, it is not immediately clear how it can be adapted in the context of other MTDs. A clear idea of when such scenarios arise and finding ways to address them would constitute an important line of research in the future.

A key idea is to have a continuous feedback cycle that verifies the security policy in place post MTD-countermeasure deployment. This can be done by ensuring end-to-end integration and regression test for the various use-cases pertaining to network traffic. Another solution could be to incorporate the policy conflicts that might arise into the modeling of the MTD. This would produce *safe* movement policies that forsee the use of MTD as a new attack surface and avoid policy conflicts.

E. Proposing novel Moving Target Defenses

An important goal of this survey is to establish a common terminology for MTD researchers. Thus, we try to categorize an array of existing works in this terminology. For example, consider the work [79]. This MTD can be categorized as a moving target defense for the movement of detection surfaces with fixed interval switching formulated as a multi-stage game that performs simulation studies of simple use-cases and measures the security and performance of individual defenses in these settings.

An interesting idea would be to turn this goal of ours on its head and explore the design of new MTDs based on the permutations of the various categorization aspects designed in this survey. For example, a hybrid surface shifting MTD that (1) shifts the detection surface and then based on a stochastic environment, shifts the prevention surface, (2) models this problem as a two-step game, (3) considers rewards that incorporate performance of individual actions and security of the ensemble, and (4) showcases experimental results on an emulated testbed is a novel Moving Target Defense.

VII. Conclusion

In this survey, we looked at various Moving Target Defenses (MTDs) that have been proposed for enhancing network security. First, we categorized them based on *what* surfaces these defenses move (the configuration set), *when* a move operation occurs (the timing function), and *how* they move between the different constituent system configurations (the movement function). In doing so, we highlight how the movement of particular software surfaces is linked to Advanced

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

Persistent Threats; thereby, allowing us to understand how the various MTDs can help thwart real-world sophisticated attacks. The dearth of works that consider the simultaneous movement of different surfaces points to an exciting research direction and possibly, the invention of more effective MTDs. In answering *how to move*, we notice that many approaches leverage Artificial Intelligence (AI) methods in general and game-theoretic techniques in particular for crafting intelligent movement strategies.

Second, we discuss how these MTDs can be implemented in practice. We find that the use of centralized technologies like Software Defined Networking (SDN) helps in implementing the MTD countermeasures with limited impact on network performance. We showcase how the surveyed MTDs are implemented in the context of real-world systems ranging from simulation studies to use in commercial products. We highlight the key technologies leveraged by the various MTDs, the layers of the network protocol stack at which an MTD is effective and the level of maturity at which it is implemented. We briefly described a few test-beds that either have been leveraged by existing MTDs and encourage researchers to use them for evaluating the effectiveness of a proposed MTD. We conclude that (1) SDN/NFV is a dominant technology used by MTDs and (2) industrial adoption of MTD solutions is still limited to few application-security products.

Third, we discuss various metrics used for measuring the effectiveness of MTDs. We put forth two categorizations based on whether these metrics (1) consider security and performance impact of the designed system and (2) are used in the modeling phase to generate particular behavior *vs.* used for evaluation of the MTD. We notice that none of the proposed MTDs consider all the metrics we put forth. One wonders if a defense that models all the proposed metrics can be realized in practice and if so, prove to be better that existing defenses both in terms of performance and security.

Lastly, we highlight areas of network security where the scope of developing MTDs hasn't been investigated. We believe that coming up with defenses which can fill these gaps will improve security and/or performance aspects of various network systems. We conclude by showcasing how our categorization provides a common terminology for researchers to describe existing MTDs and develop future ones.

VIII. Acknowledgement

This research is supported in part by following research grants: Naval Research Lab N00173-15-G017, AFOSR grant FA9550-18-1-0067, the NASA grant NNX17AD06G, ONR grants N00014-16-1-2892, N00014-18-1-2442, N00014-18-12840, NSFUS DGE-1723440, OAC-1642031, SaTC-1528099, 1723440 and NSFChina 61628201 and 61571375 and a JP Morgan AI Research Faculty Award. Sailik Sengupta is supported by the IBM Ph.D. Fellowship. Also, Abdulhakim Sabur is a scholarship recipient from Taibah University through Saudi Arabian Cultural Mission (SACM).

References

[1] ESDS, "Gartner Identifies the Top 10 Trends Impacting Infrastructure and Operations for 2019," https://www.gartner.com/en/newsroom/press-releases/2018-12-04-gartner-identifies-the-top-10-trends-impacting-infras, 2018, online; accessed 03 Marsh 2020.

[2] Kasey Panetta, "Gartners Top 10 Security Predictions 2016," https://www.gartner.com/smarterwithgartner/top-10-security-predictions-2016/, 2016, online; accessed 11 Nov 2018.

[3] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[4] D. Kreutz, F. M. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[5] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu, "A survey of game theory as applied to network security," in *2010 43rd Hawaii International Conference on System Sciences*. IEEE, 2010, pp. 1–10.

[6] H. Okhravi, M. Rabe, T. Mayberry, W. Leonard, T. Hobson, D. Bigelow, and W. Streilein, "Survey of cyber moving target techniques," MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB, Tech. Rep., 2013.

[7] H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein, "Finding focus in the blur of moving-target techniques," *IEEE Security & Privacy*, vol. 12, no. 2, pp. 16–26, 2014.

[8] K. A. Farris and G. Cybenko, "Quantification of moving target cyber defenses," in *Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security, Defense, and Law Enforcement XIV*, vol. 9456. International Society for Optics and Photonics, 2015, p. 94560L.

[9] G.-l. Cai, B.-s. Wang, W. Hu, and T.-z. Wang, "Moving target defense: state of the art and characteristics," *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 11, pp. 1122–1153, 2016.

[10] C. Lei, H.-Q. Zhang, J.-L. Tan, Y.-C. Zhang, and X.-H. Liu, "Moving target defense techniques: A survey," *Security and Communication Networks*, vol. 2018, 2018.

[11] O. S. V. D. (OSVDB), "Open source vulnerability database (osvdb)," 2012.

[12] C. V. CVE, "Exposures," 2012.

[13] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities," *IEEE Communications Surveys & Tutorials*, 2019.

[14] P. Chen, L. Desmet, and C. Huygens, "A study on advanced persistent threats," in *IFIP International Conference on Communications and Multimedia Security*. Springer, 2014, pp. 63–72.

[15] L. Martin, "Cyber kill chain (ckc)," https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html, 2017, online; accessed 11 Nov 2018.

[16] M. I. Center, "Apt1: Exposing one of chinas cyber espionage units," *Mandian. com*, 2013.

[17] I. Friedberg, F. Skopik, G. Settanni, and R. Fiedler, "Combating advanced persistent threats: From network event correlation to incident detection," *Computers & Security*, vol. 48, pp. 35–57, 2015.

[18] D. Moon, H. Im, J. Lee, and J. Park, "Mlds: multi-layer defense system for preventing advanced persistent threats," *Symmetry*, vol. 6, no. 4, pp. 997–1010, 2014.

[19] R. Kissel, *Glossary of key information security terms*. Diane Publishing, 2011.

[20] K. Glass and R. Colbaugh, "Web analytics for security informatics," in *Intelligence and Security Informatics Conference (EISIC), 2011 European*. IEEE, 2011, pp. 214–219.

[21] P. Mehra, "A brief study and comparison of snort and bro open source network intrusion detection systems," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 1, no. 6, pp. 383–386, 2012.

[22] R. U. Rehman, *Intrusion detection systems with Snort: advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID*. Prentice Hall Professional, 2003.

[23] P. Deshpande, S. Sharma, S. Peddoju, and S. Junaid, "Hids: A host based intrusion detection system for cloud computing environment," *International Journal of System Assurance Engineering and Management*, vol. 9, no. 3, pp. 567–576, 2018.

[24] G. H. Kim and E. H. Spafford, "The design and implementation of tripwire: A file system integrity checker," in *Proceedings of the 2nd ACM Conference on Computer and Communications Security*. ACM, 1994, pp. 18–29.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

[25] X. Zhang, C. Li, and W. Zheng, "Intrusion prevention system design," in *null*. IEEE, 2004, pp. 386–390.

[26] H. Welte, "The netfilter framework in linux 2.4," in *Proceedings of Linux Kongress*, 2000.

[27] F. Pouget, M. Dacier *et al.*, "Honeypot-based forensics," in *AusCERT Asia Pacific Information Technology Security Conference*, 2004.

[28] M. Oosterhof, "Cowrie honeypot," 2014.

[29] SANS, "Intrusion detection evasion: How attackers get past the burglar alarm," https://www.sans.org/reading-room/whitepapers/detection/intrusion-detection-evasion-attackers-burglar-alarm-1284, 2003, online; accessed 11 Nov 2018.

[30] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[31] A. Al-Dujaili, A. Huang, E. Hemberg, and U.-M. OReilly, "Adversarial deep learning for robust detection of binary encoded malware," in *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2018, pp. 76–82.

[32] N. Stojanovski, M. Gusev, D. Gligoroski, and S. J. Knapskog, "Bypassing data execution prevention on microsoftwindows xp sp2," in *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*. IEEE, 2007, pp. 1222–1226.

[33] Z. Shu and G. Yan, "Ensuring deception consistency for ftp services hardened against advanced persistent threats," in *Proceedings of the 5th ACM Workshop on Moving Target Defense*. ACM, 2018, pp. 69–79.

[34] S. H. Houmb, V. N. Franqueira, and E. A. Engum, "Quantifying security risk level from cvss estimates of frequency and impact," *JSS*, vol. 83, no. 9, pp. 1622–1634, 2010.

[35] B. Schneier, "Attack trees," *Dr. Dobbs journal*, vol. 24, no. 12, pp. 21–29, 1999.

[36] S. Mauw and M. Oostdijk, "Foundations of attack trees," in *Icisc*, vol. 3935. Springer, 2005, pp. 186–198.

[37] K. Ingols, R. Lippmann, and K. Piwowarski, "Practical attack graph generation for network defense," in *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual*. IEEE, 2006, pp. 121–130.

[38] M. Albanese, S. Jajodia, and S. Noel, "Time-efficient and cost-effective network hardening using attack graphs," in *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*. IEEE, 2012, pp. 1–12.

[39] S. Jha, O. Sheyner, and J. Wing, "Two formal analyses of attack graphs," in *Computer Security Foundations Workshop, 2002. Proceedings. 15th IEEE*. IEEE, 2002, pp. 49–63.

[40] X. Ou, W. F. Boyer, and M. A. McQueen, "A scalable approach to attack graph generation," in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 336–345.

[41] J. Lee, H. Lee, and H. P. In, "Scalable attack graph for risk assessment," in *Information Networking, 2009. ICOIN 2009. International Conference on*. IEEE, 2009, pp. 1–5.

[42] J. Homer, X. Ou, and M. A. McQueen, "From attack graphs to automated configuration managementan iterative approach," *Kansas State University Technical Report*, 2008.

[43] R. E. Sawilla and X. Ou, "Identifying critical attack assets in dependency attack graphs," in *European Symposium on Research in Computer Security*. Springer, 2008, pp. 18–34.

[44] H. Huang, S. Zhang, X. Ou, A. Prakash, and K. Sakallah, "Distilling critical attack graph surface iteratively through minimum-cost sat solving," in *Proceedings of the 27th Annual Computer Security Applications Conference*. ACM, 2011, pp. 31–40.

[45] O. Sheyner and J. Wing, "Tools for generating and analyzing attack graphs," in *International Symposium on Formal Methods for Components and Objects*. Springer, 2003, pp. 344–371.

[46] J. B. Hong and D. S. Kim, "Scalable security model generation and analysis using k-importance measures," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2013, pp. 270–287.

[47] J. B. Hong, D. S. Kim, C.-J. Chung, and D. Huang, "A survey on the usability and practical applications of graphical security models," *Computer Science Review*, vol. 26, pp. 1–16, 2017.

[48] P. Ammann, D. Wijesekera, and S. Kaushik, "Scalable, graph-based network vulnerability analysis," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM, 2002, pp. 217–224.

[49] X. Ou, S. Govindavajhala, and A. W. Appel, "Mulval: A logic-based network security analyzer." in *USENIX Security Symposium*. Baltimore, MD, 2005, pp. 8–8.

[50] J. B. Hong and D. S. Kim, "Performance analysis of scalable attack representation models," in *IFIP International Information Security Conference*. Springer, 2013, pp. 330–343.

[51] A. Chowdhary, S. Pisharody, and D. Huang, "Sdn based scalable mtd solution in cloud network," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 27–36.

[52] E. Al-Shaer, Q. Duan, and J. H. Jafarian, "Random host mutation for moving target defense," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2012, pp. 310–327.

[53] M. Albanese, A. De Benedictis, S. Jajodia, and K. Sun, "A moving target defense mechanism for manets based on identity virtualization," in *2013 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2013, pp. 278–286.

[54] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: transparent moving target defense using software defined networking," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 127–132.

[55] A. Aydeger, N. Saputro, K. Akkaya, and M. Rahman, "Mitigating crossfire attacks using sdn-based moving target defense," in *Local Computer Networks (LCN), 2016 IEEE 41st Conference on*. IEEE, 2016, pp. 627–630.

[56] Z. Zhao, F. Liu, and D. Gong, "An sdn-based fingerprint hopping method to prevent fingerprinting attacks," *Security and Communication Networks*, vol. 2017, 2017.

[57] A. Schlenker, O. Thakoor, H. Xu, F. Fang, M. Tambe, L. Tran-Thanh, P. Vayanos, and Y. Vorobeychik, "Deceiving cyber adversaries: A game theoretic approach," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 892–900.

[58] S. Jajodia, N. Park, E. Serra, and V. Subrahmanian, "Share: A stackelberg honey-based adversarial reasoning engine," *ACM Transactions on Internet Technology (TOIT)*, vol. 18, no. 3, p. 30, 2018.

[59] R. Algin, H. O. Tan, and K. Akkaya, "Mitigating selective jamming attacks in smart meter data collection using moving target defense," in *Proceedings of the 13th ACM Symposium on QoS and Security for Wireless and Mobile Networks*. ACM, 2017, pp. 1–8.

[60] C. Lei, D.-H. Ma, and H.-Q. Zhang, "Optimal strategy selection for moving target defense based on markov game," *IEEE Access*, vol. 5, pp. 156–169, 2017.

[61] R. Zhuang, S. A. DeLoach, and X. Ou, "Towards a theory of moving target defense," in *Proceedings of the First ACM Workshop on Moving Target Defense*. ACM, 2014, pp. 31–40.

[62] P. K. Manadhata, "Game theoretic approaches to attack surface shifting," in *Moving Target Defense II*. Springer, 2013, pp. 1–13.

[63] Q. Zhu and T. Başar, "Game-theoretic approach to feedback-driven multi-stage moving target defense," in *International Conference on Decision and Game Theory for Security*. Springer, 2013, pp. 246–263.

[64] S. Sengupta, S. G. Vadlamudi, S. Kambhampati, A. Doupé, Z. Zhao, M. Taguinod, and G.-J. Ahn, "A game theoretic approach to strategy generation for moving target defense in web applications," in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2017, pp. 178–186.

[65] K. M. Carter, J. F. Riordan, and H. Okhravi, "A game theoretic approach to strategy determination for dynamic platform defenses," in *Proceedings of the First ACM Workshop on Moving Target Defense*. ACM, 2014, pp. 21–30.

[66] M. Thompson, N. Evans, and V. Kisekka, "Multiple os rotational environment an implemented moving target defense," in *Resilient Control Systems (ISRCS), 2014 7th International Symposium on*. IEEE, 2014, pp. 1–6.

[67] I. El Mir, A. Chowdhary, D. Huang, S. Pisharody, D. S. Kim, and A. Haqiq, "Software defined stochastic model for moving target defense," in *International Afro-European Conference for Industrial Advancement*. Springer, 2016, pp. 188–197.

[68] S. Debroy, P. Calyam, M. Nguyen, A. Stage, and V. Georgiev, "Frequency-minimal moving target defense using software-defined networking," in *2016 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2016, pp. 1–6.

[69] A. Prakash and M. P. Wellman, "Empirical game-theoretic analysis for moving target defense," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 57–65.

[70] S. Neti, A. Somayaji, and M. E. Locasto, "Software diversity: Security, entropy and game theory."

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

[71] M. Crouse, E. W. Fulp, and D. Canas, "Improving the diversity defense of genetic algorithm-based moving target approaches," in *Proceedings of the National Symposium on Moving Target Research*, 2012.

[72] B. Bohara, "Moving target defense using live migration of docker containers," Ph.D. dissertation, Arizona State University, 2017.

[73] A. Chowdhary, S. Pisharody, A. Alshamrani, and D. Huang, "Dynamic game based security framework in sdn-enabled cloud networking environments," in *Proceedings of the ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*. ACM, 2017, pp. 53–58.

[74] A. Clark, K. Sun, L. Bushnell, and R. Poovendran, "A game-theoretic approach to ip address randomization in decoy-based cyber defense," in *International Conference on Decision and Game Theory for Security*. Springer, 2015, pp. 3–21.

[75] R. Colbaugh and K. Glass, "Predictability-oriented defense against adaptive adversaries," in *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2721–2727.

[76] S. Venkatesan, M. Albanese, G. Cybenko, and S. Jajodia, "A moving target defense approach to disrupting stealthy botnets," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 37–46.

[77] S. Sengupta, A. Chowdhary, D. Huang, and S. Kambhampati, "Moving target defense for the placement of intrusion detection systems in the cloud," in *International Conference on Decision and Game Theory for Security*. Springer, 2018, pp. 326–345.

[78] ——, "General sum markov games for strategic detection of advanced persistent threats using moving target defense in cloud networks," 2019.

[79] A. Chowdhary*, S. Sengupta*, D. Huang, and S. Kambhampati, "Markov game modeling of moving target defense for strategic detection of threats in cloud networks," *AAAI Workshop on Artificial Intelligence for Cyber Security (AICS)*, 2019.

[80] S. Sengupta, T. Chakraborti, and S. Kambhampati, "Mtdeep: boosting the security of deep neural nets against adversarial attacks with moving target defense," in *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[81] A. Sinha, T. H. Nguyen, D. Kar, M. Brown, M. Tambe, and A. X. Jiang, "From physical security to cybersecurity," *Journal of Cybersecurity*, vol. 1, no. 1, pp. 19–35, 2015.

[82] T. Salman, D. Bhamare, A. Erbad, R. Jain, and M. Samaka, "Machine learning for anomaly detection and categorization in multi-cloud environments," in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. IEEE, 2017, pp. 97–103.

[83] Y. Vorobeychik and B. Li, "Optimal randomized classification in adversarial settings," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 485–492.

[84] D. Bhamare, T. Salman, M. Samaka, A. Erbad, and R. Jain, "Feasibility of supervised machine learning for cloud security," in *2016 International Conference on Information Science and Security (ICISS)*. IEEE, 2016, pp. 1–5.

[85] M. Van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "Flipit: The game of stealthy takeover," *Journal of Cryptology*, vol. 26, no. 4, pp. 655–713, 2013.

[86] Y. Shi, H. Zhang, J. Wang, F. Xiao, J. Huang, D. Zha, H. Hu, F. Yan, and B. Zhao, "Chaos: An sdn-based moving target defense system," *Security and Communication Networks*, vol. 2017, 2017.

[87] I. El Mir, A. Chowdhary, D. Huang, S. Pisharody, D. S. Kim, and A. Haqiq, "Software defined stochastic model for moving target defense," in *International Afro-European Conference for Industrial Advancement*. Springer, 2016, pp. 188–197.

[88] C.-J. Chung, P. Khatkar, T. Xing, J. Lee, and D. Huang, "Nice: Network intrusion detection and countermeasure selection in virtual network systems," *IEEE transactions on dependable and secure computing*, vol. 10, no. 4, pp. 198–211, 2013.

[89] A. Chowdhary, A. Alshamrani, D. Huang, and H. Liang, "Mtd analysis and evaluation framework in software defined network (mason)," in *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*. ACM, 2018, pp. 43–48.

[90] G. F. Lyon, *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure, 2009.

[91] V. N. Padmanabhan and D. R. Simon, "Secure traceroute to detect faulty or malicious routing," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 77–82, 2003.

[92] S. G. Vadlamudi, S. Sengupta, M. Taguinod, Z. Zhao, A. Doupé, G.-J. Ahn, and S. Kambhampati, "Moving target defense for web applications using bayesian stackelberg games," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 1377–1378.

[93] A. Chowdhary, S. Sengupta, A. Alshamrani, D. Huang, and A. Sabur, "Adaptive mtd security using markov game modeling," in *2019 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2019, pp. 577–581.

[94] H. Maleki, S. Valizadeh, W. Koch, A. Bestavros, and M. van Dijk, "Markov modeling of moving target defense games," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 81–92.

[95] E. Miehling, M. Rasouli, and D. Teneketzis, "Optimal defense policies for partially observable spreading processes on bayesian attack graphs," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 67–76.

[96] T. H. Nguyen, M. Wright, M. P. Wellman, and S. Singh, "Multistage attack graph security games: Heuristic strategies, with empirical game-theoretic analysis," *Security and Communication Networks*, vol. 2018, 2018.

[97] L. S. Shapley, "Stochastic games," *Proceedings of the national academy of sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.

[98] M. J. Osborne *et al.*, *An introduction to game theory*. Oxford university press New York, 2004, vol. 3, no. 3.

[99] S. Wang, H. Shi, Q. Hu, B. Lin, and X. Cheng, "Moving target defense for internet of things based on the zero-determinant theory," *IEEE Internet of Things Journal*, 2019.

[100] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[101] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, "Dynamic programming for partially observable stochastic games," in *AAAI*, vol. 4, 2004, pp. 709–715.

[102] S. Valizadeh and M. van Dijk, "Toward a theory of cyber attacks," *arXiv preprint arXiv:1901.01598*, 2019.

[103] H. Li, W. Shen, and Z. Zheng, "Spatial-temporal moving target defense: A markov stackelberg game model," *arXiv preprint arXiv:2002.10390*, 2020.

[104] J. Sherry, S. Ratnasamy, and J. S. At, "A survey of enterprise middlebox deployments," 2012.

[105] J. Steinberger, B. Kuhnert, C. Dietz, L. Ball, A. Sperotto, H. Baier, A. Pras, and G. Dreo, "Ddos defense using mtd and sdn."

[106] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow *et al.*, "Onos: towards an open, distributed sdn os," in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.

[107] Polyverse, "Moving target defense: Redefining the power of defense," https://view.attach.io/ByfWW3KGf, 2018, online; accessed 11 Feb 2019.

[108] TrapX, "Deceptiongrid trapx," http://trapx.com/wp-content/uploads/2018/03/, 2018, online; accessed 11 Feb 2019.

[109] CryptoTrap, "Cryptotrap trapx," http://trapx.com/landing/wp-content/uploads/2018/12/Product_Brief_TrapX_CryptoTrap.pdf, 2018, online; accessed 11 Feb 2019.

[110] J. B. Hong, S. Yoon, H. Lim, and D. S. Kim, "Optimal network reconfiguration for software defined networks using shuffle-based online mtd," in *Reliable Distributed Systems (SRDS), 2017 IEEE 36th Symposium on*. IEEE, 2017, pp. 234–243.

[111] J. Medved, R. Varga, A. Tkacik, and K. Gray, "Opendaylight: Towards a model-driven sdn controller architecture," in *2014 IEEE 15th International Symposium on*. IEEE, 2014, pp. 1–6.

[112] Y. Wang, Q. Chen, J. Yi, and J. Guo, "U-tri: unlinkability through random identifier for sdn network," in *Proceedings of the 2017 Workshop on Moving Target Defense*. ACM, 2017, pp. 3–15.

[113] OSRG, "Ryu sdn controller," https://osrg.github.io/ryu/, 2017.

[114] A. Homescu, T. Jackson, S. Crane, S. Brunthaler, P. Larsen, and M. Franz, "Large-scale automated software diversityprogram evolution redux," *IEEE Transactions on Dependable and Secure Computing*, no. 1, pp. 1–1.

[115] C. Lattner and V. Adve, "The llvm compiler framework and infrastructure tutorial," in *International Workshop on Languages and Compilers for Parallel Computing*. Springer, 2004, pp. 15–16.

[116] X. Han, N. Kheir, and D. Balzarotti, "Evaluation of deception-based web attacks detection." in *MTD@ CCS*, 2017, pp. 65–73.

[117] S. Baxter and L. C. Vogt, "Content management system," Mar. 12 2002, uS Patent 6,356,903.

[118] W. Connell, D. A. Menascé, and M. Albanese, "Performance modeling of moving target defenses," in *Proceedings of the 2017 Workshop on Moving Target Defense*. ACM, 2017, pp. 53–63.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

[119] S. Scherfke and O. Lnsdorf, "Simpy testbed," https://simpy.readthedocs.io/en/3.0/.

[120] R. Ricci, E. Eide, and C. Team, "Introducing cloudlab: Scientific infrastructure for advancing cloud architectures and applications," *; login:: the magazine of USENIX & SAGE*, vol. 39, no. 6, pp. 36–38, 2014.

[121] A. C. Pappa, "Moving target defense for securing smart grid communications: Architectural design, implementation and evaluation," Ph.D. dissertation, Iowa State University, 2016.

[122] Morphisec, "Moving target defense," https://www.morphisec.com/resources, 2018, online; accessed 12 Feb 2019.

[123] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "Openstack: toward an open-source solution for cloud computing," *International Journal of Computer Applications*, vol. 55, no. 3, pp. 38–42, 2012.

[124] N. O. Ahmed and B. Bhargava, "Mayflies: A moving target defense framework for distributed systems," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 59–64.

[125] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "Geni: A federated testbed for innovative network experiments," *Computer Networks*, vol. 61, pp. 5–23, 2014.

[126] J. Taylor, K. Zaffarano, B. Koller, C. Bancroft, and J. Syversen, "Automated effectiveness evaluation of moving target defenses: metrics for missions and attacks," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 129–134.

[127] J. Yackoski, H. Bullen, X. Yu, and J. Li, "Applying self-shielding dynamics to the network architecture," in *Moving target defense II*. Springer, 2013, pp. 97–115.

[128] J. Yackoski, J. Li, S. A. DeLoach, and X. Ou, "Mission-oriented moving target defense based on cryptographically strong network dynamics," in *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*. ACM, 2013, p. 57.

[129] J. Yackoski, P. Xie, H. Bullen, J. Li, and K. Sun, "A self-shielding dynamic network architecture," in *2011-MILCOM 2011 Military Communications Conference*. IEEE, 2011, pp. 1381–1386.

[130] R. Chadha, T. Bowen, C.-Y. J. Chiang, Y. M. Gottlieb, A. Poylisher, A. Sapello, C. Serban, S. Sugrim, G. Walther, L. M. Marvel *et al.*, "Cybervan: A cyber security virtual assured network testbed," in *Military Communications Conference, MILCOM 2016-2016 IEEE*. IEEE, 2016, pp. 1125–1130.

[131] K. Zaffarano, J. Taylor, and S. Hamilton, "A quantitative framework for moving target defense effectiveness evaluation," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 3–10.

[132] Y.-B. Luo, B.-S. Wang, X.-F. Wang, X.-F. Hu, G.-L. Cai, and H. Sun, "Rpah: Random port and address hopping for thwarting internal and external adversaries," in *2015 IEEE Trustcom/BigDataSE/ISPA*, vol. 1. IEEE, 2015, pp. 263–270.

[133] P. Kampanakis, H. Perros, and T. Beyene, "Sdn-based solutions for moving target defense network protection," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*. IEEE, 2014, pp. 1–6.

[134] Q. Jia, K. Sun, and A. Stavrou, "Motag: Moving target defense against internet denial of service attacks," in *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*. IEEE, 2013, pp. 1–9.

[135] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: an overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, 2003.

[136] L. Dhanabal and S. Shantharajah, "A study on nsl-kdd dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[137] M. Dunlop, S. Groat, W. Urbanski, R. Marchany, and J. Tront, "Mt6d: A moving target ipv6 defense," in *2011-MILCOM 2011 Military Communications Conference*. IEEE, 2011, pp. 1321–1326.

[138] S. Thomson, T. Narten, and T. Jinmei, "Ipv6 stateless address autoconfiguration," Tech. Rep., 2007.

[139] E. Al-Shaer, "Toward network configuration randomization for moving target defense," in *Moving Target Defense*. Springer, 2011, pp. 153–159.

[140] J. H. Cox, J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley, and H. L. Owen, "Advancing software-defined networks: A survey," *IEEE Access*, vol. 5, pp. 25 487–25 526, 2017.

[141] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a globally-deployed software defined wan," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 3–14.

[142] S. T. Ali, V. Sivaraman, A. Radford, and S. Jha, "A survey of securing networks using software defined networking." *IEEE Trans. Reliability*, vol. 64, no. 3, pp. 1086–1097, 2015.

[143] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *Lisa*, vol. 99, no. 1, 1999, pp. 229–238.

[144] D. Fudenberg, D. Levine, and E. Maskin, "The folk theorem with imperfect public information," in *A Long-Run Collaboration On Long-Run Games*. World Scientific, 2009, pp. 231–273.

[145] R. Zhuang, S. Zhang, A. Bardas, S. A. DeLoach, X. Ou, and A. Singhal, "Investigating the application of moving target defenses to network security," in *Resilient Control Systems (ISRCS), 2013 6th International Symposium on*. IEEE, 2013, pp. 162–169.

[146] A. Kosowski and V. Mosorov, "Nessi2 simulator for large-scale ddos attack analysis," in *Perspective Technologies and Methods in MEMS Design*. IEEE, 2011, pp. 157–159.

[147] A. G. Bardas, S. C. Sundaramurthy, X. Ou, and S. A. DeLoach, "Mtd cbits: Moving target defense for cloud-based it systems," in *European Symposium on Research in Computer Security*. Springer, 2017, pp. 167–186.

[148] T. R. Henderson, M. Lacage, G. F. Riley, C. Dowell, and J. Kopena, "Network simulations with the ns-3 simulator," *SIGCOMM demonstration*, vol. 14, no. 14, p. 527, 2008.

[149] M. Carvalho, J. DeMott, R. Ford, and D. A. Wheeler, "Heartbleed 101," *IEEE security & privacy*, vol. 12, no. 4, pp. 63–67, 2014.

[150] R. Meier, P. Tsankov, V. Lenders, L. Vanbever, and M. Vechev, "Nethide: secure and practical network topology obfuscation," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 693–709.

[151] S. Rass, S. König, and S. Schauer, "Defending against advanced persistent threats using game-theory," *PloS one*, vol. 12, no. 1, p. e0168675, 2017.

[152] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[153] M. Crouse, B. Prosser, and E. W. Fulp, "Probabilistic performance analysis of moving target and deception reconnaissance defenses," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 21–29.

[154] H. Wang, F. Li, and S. Chen, "Towards cost-effective moving target defense against ddos and covert channel attacks," in *Proceedings of the 2016 ACM Workshop on Moving Target Defense*. ACM, 2016, pp. 15–25.

[155] C. Lei, D.-h. Ma, H.-q. Zhang, and L.-m. Wang, "Moving target network defense effectiveness evaluation based on change-point detection," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[156] H. Alavizadeh, J. Jang-Jaccard, and D. S. Kim, "Evaluation for combination of shuffle and diversity on moving target defense strategy for cloud computing," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2018, pp. 573–578.

[157] S. Pisharody, J. Natarajan, A. Chowdhary, A. Alshalan, and D. Huang, "Brew: A security policy analysis framework for distributed sdn-based cloud environments," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[158] S. Pisharody, A. Chowdhary, and D. Huang, "Security policy checking in distributed sdn based clouds," in *Communications and Network Security (CNS), 2016 IEEE Conference on*. IEEE, 2016, pp. 19–27.

[159] H. Hamed and E. Al-Shaer, "Taxonomy of conflicts in network security policies," *IEEE Communications Magazine*, vol. 44, no. 3, pp. 134–141, 2006.

[160] A. Khurshid, X. Zou, W. Zhou, M. Caesar, and P. B. Godfrey, "Veriflow: Verifying network-wide invariants in real time," in *Presented as part of the 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, 2013, pp. 15–27.

[161] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia, "An attack graph-based probabilistic security metric," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2008, pp. 283–296.

[162] A. Roy, D. S. Kim, and K. S. Trivedi, "Attack countermeasure trees (act): towards unifying the constructs of attack and defense trees," *Security and Communication Networks*, vol. 5, no. 8, pp. 929–943, 2012.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2020.2982955, IEEE Communications Surveys & Tutorials

[163] J. B. Hong and D. S. Kim, "Assessing the effectiveness of moving target defenses using security models," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 163–177, 2016.

[164] M. Schiffman and C. Cisco, "A complete guide to the common vulnerability scoring system (cvss)," *White paper. Identification of Basic Measurable Security Components in Software Intensive Systems*, 2005.

[165] M. Christodorescu, M. Fredrikson, S. Jha, and J. Giffin, "End-to-end software diversification of internet services," in *Moving Target Defense*. Springer, 2011, pp. 117–130.

[166] R. Zhuang, S. Zhang, S. A. DeLoach, X. Ou, and A. Singhal, "Simulation-based approaches to studying effectiveness of moving-target network defense," in *National symposium on moving target research*, vol. 246, 2012.

[167] J. Xu, P. Guo, M. Zhao, R. F. Erbacher, M. Zhu, and P. Liu, "Comparing different moving target defense techniques," in *Proceedings of the First ACM Workshop on Moving Target Defense*. ACM, 2014, pp. 97–107.

[168] J. B. Hong and D. S. Kim, "Scalable security models for assessing effectiveness of moving target defenses," in *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on*. IEEE, 2014, pp. 515–526.

[169] S. Rass and Q. Zhu, "Gadapt: A sequential game-theoretic framework for designing defense-in-depth strategies against advanced persistent threats," in *International Conference on Decision and Game Theory for Security*. Springer, 2016, pp. 314–326.

[170] A. Sabur, A. Chowdhary, D. Huang, M. Kang, A. Kim, and A. Velazquez, "S3: A dfw-based scalable security state analysis framework for large-scale data center networks," in *22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019)*, 2019, pp. 473–485.

[171] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[172] N. Spring, R. Mahajan, and D. Wetherall, "Measuring isp topologies with rocketfuel," in *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4. ACM, 2002, pp. 133–145.

[173] J. Frahim, O. Santos, and A. Ossipov, *Cisco ASA: All-in-one Next-Generation Firewall, IPS, and VPN Services*. Cisco Press, 2014.

[174] P. Alto, "Palo alto next generation firewall," 2018.

[175] O. Mämmelä, J. Hiltunen, J. Suomalainen, K. Ahola, P. Mannersalo, and J. Vehkaperä, "Towards micro-segmentation in 5g network security," in *European Conference on Networks and Communications (EuCNC 2016) Workshop on Network Management, Quality of Service and Security for 5G Networks*, 2016.

Sailik Sengupta is a Ph.D. student in Computer Science at Arizona State University. He received his B.E. in Computer Science and Engineering from Jadavpur University (2013) and then worked for Amazon where he became an application-security certifier. Sailik is interested in solving problems that arise in multi-agent settings where the participating agents are non-cooperative or adversarial. He has looked at challenges in cyber-security, adversarial machine learning, and human-AI interaction. Sailik was awarded the IBM Ph.D. fellowship in 2018.

Ankur Chowdhary is a Ph.D. candidate in Computer Science at Arizona State University, Tempe, AZ, USA. He received B.Tech in Information Technology from GGSIPU in 2011 and MS in Computer Science from ASU in 2015. He has worked as Information Security Researcher for Blackberry Ltd., RSG and Application Developer for CSC Pvt. Ltd. His research interests include SDN, Web Security, Network Security, and application of AI and Machine Learning in the field of Security.

Abdulhakim Sabur is a Ph.D student in Computer Engineering at Arizona state University, Tempe, AZ, USA. He received his B.S. degree (with Honor) in Computer Science and Engineering from King Saud University, Saudi Arabia in 2015 and a Master degree in Computer Engineering from Arizona State University in 2018. He worked as a researcher in King Abdulaziz City for Science and Technology (KACST) and a teaching assistant in Taibah University. His research interest include Network and information security, vulnerability analysis and management, automated policy and security checking in software defined networking systems.

Adel Alshamrani is an assistant professor in department of cybersecurity, College of Computer Science and Engineering at University of Jeddah, Jeddah, Saudi Arabia. He received his B.S. degree in computer science from Umm Al-Qura University, Saudi Arabia in 2007, M.S. degree in computer science from La Trobe University Melbourne, Australia, in 2010, and PhD in computer science from Arizona State University in 2018. He has eight years of work experience in information security, network engineering, and teaching while working in the Faculty of Computing and Information Technology, King Abdul Aziz University, and University of Jeddah. His research interests include information security, intrusion detection, and software defined networking. He is the Chief Information Security Officer (CISO) at the University of Jeddah.

Dijiang Huang received the B.S. degree from Beijing University of Posts and Telecommunications, China, and the M.S. and Ph.D. degrees from the University of Missouri, Kansas in 1995, 2001, and 2004 respectively. He is an Associate Professor with the School of Computing Informatics and Decision System Engineering, Arizona State University. His research interests include computer networking, security, and privacy. He is an Associate Editor of the Journal of Network and System Management (JNSM) and the IEEE Communications Surveys and Tutorials. He has also served as the chair at multiple international conferences and workshops. His research was supported by the NSF, ONR, ARO, NATO, and Consortium of Embedded System (CES). He was the recipient of the ONR Young Investigator Program (YIP) Award.

Subbarao Kambhampati (Rao) is a professor of Computer Science at Arizona State University. He received his B.Tech. in Electrical Engineering (Electronics) from Indian Institute of Technology, Madras (1983), and M.S.(1985) and Ph.D.(1989) in Computer Science (1985,1989) from University of Maryland, College Park. Kambhampati studies fundamental problems in planning, decision making, and game theory. Kambhampati is a fellow of AAAI, AAAS and ACM, and was an NSF Young Investigator. He received multiple teaching awards, including a university last lecture recognition. Kambhampati is the past president of AAAI and was a trustee of IJCAI. He was the program chair for IJCAI 2016, ICAPS 2013, AAAI 2005 and AIPS 2000 and served on the board of directors of Partnership on AI. Kambhampati's research, as well as his views on the progress and societal impacts of AI, have been featured in multiple national and international media outlets.

1553-877X (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of Exeter. Downloaded on May 03,2020 at 15:59:26 UTC from IEEE Xplore. Restrictions apply.

Research Paper Name :-

A Survey of Moving Target Defenses for Network Security

Niranjan Joshi
Pratham Karani
Manav Jain

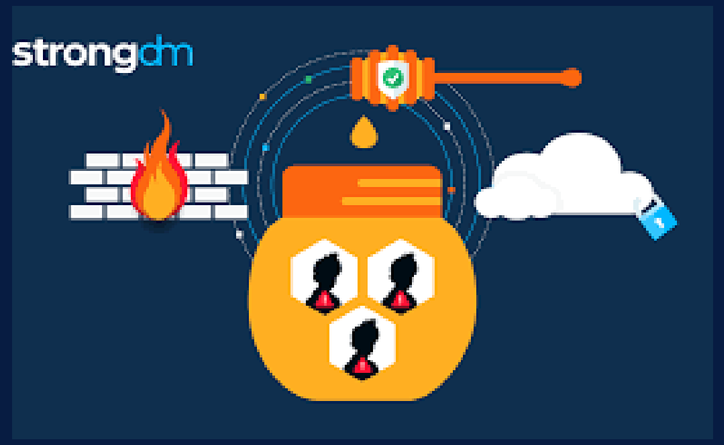
Moving Target Defense

- Constantly change locks in house and rearrange furniture so that burglar gets confused
- In the cyber world, it means regularly changing the way your computer network works to make it harder for hackers to figure out how to break in.

Abstract Idea

- Traditional defense system - static & predictable , hence easy for attacker to attack
- Moving Target Defenses - constant change in setup of system , hence difficult for attacker to attack

Different defense methods

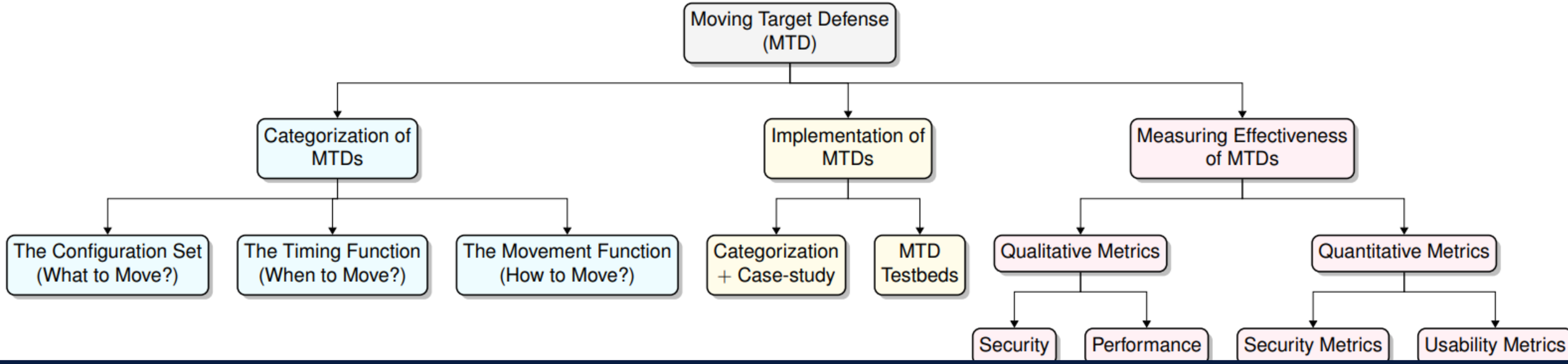


- Web Analytics
- Intrusion Detection System(IDS)
- Intrusion Prevention System(IPS)
- Proxy Filtering
- Tarpit
- HoneyPot

Why Moving Target Defenses ?

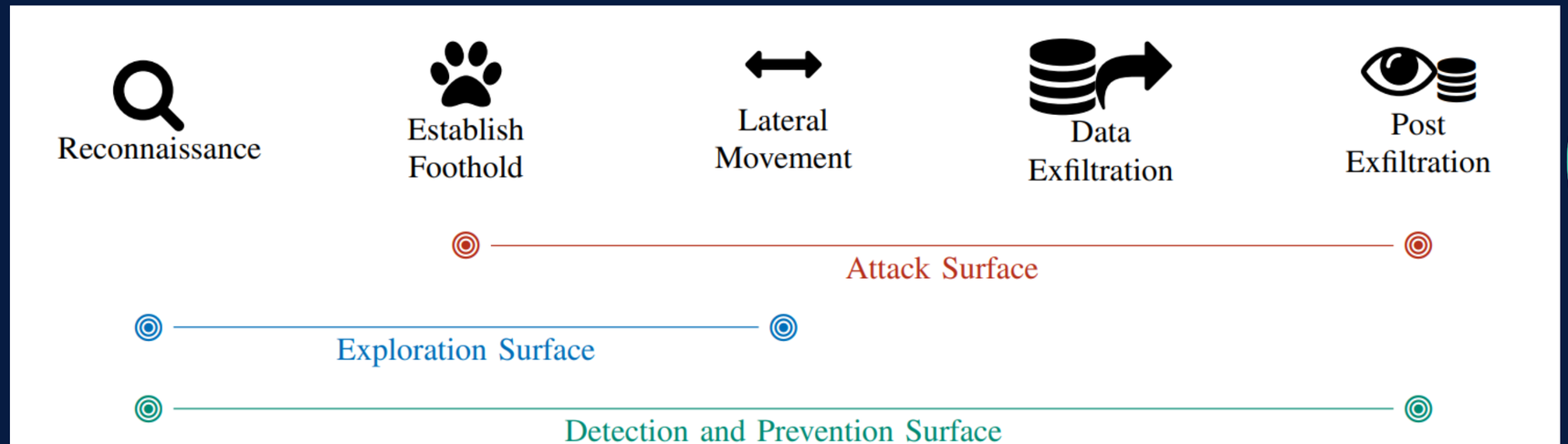
- Network and cloud infrastructure is growing complex each day , hence robust security measures are essential
- Traditional security methods such as network traffic , route patching , perimeter defense have several shortcomings
- Hence MTD is needed and along with NFV (network function virtualization) and SDN (software defined networking)

Types of MTD


Moving Target Defense (MTD)
Categorization of MTDs
Implementation of MTDs
Measuring Effectiveness of MTDs
The Configuration Set (What to Move?)
The Timing Function (When to Move?)
The Movement Function (How to Move?)
Categorization + Case-study
MTD Testbeds
Qualitative Metrics
Quantitative Metrics
Security
Performance
Security Metrics
Usability Metrics

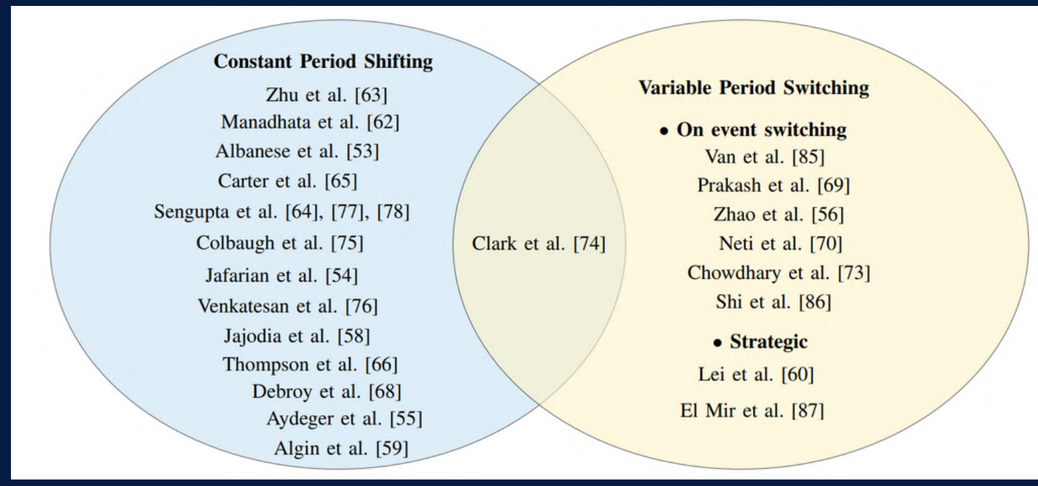
What to Move?

- Exploration Surface Shifting
- Attack Surface Shifting
- Detection Surface Shifting
- Prevention Surface Shifting
- Multi Surface Shifting



When to Move ?

- Constant Period Switching (CPS)
- Variable Period Switching (VPS)
  a) On event
  b) Strategic
  c) Hybrid



How to Move ?

- Markow game framework - State(S) , Pure Strategies (AD) , Mixed Strategies (AA) , Transitions (t) , Reward (RD , RA)

1) Single Stage Modelling
2) Multi Stage Modelling

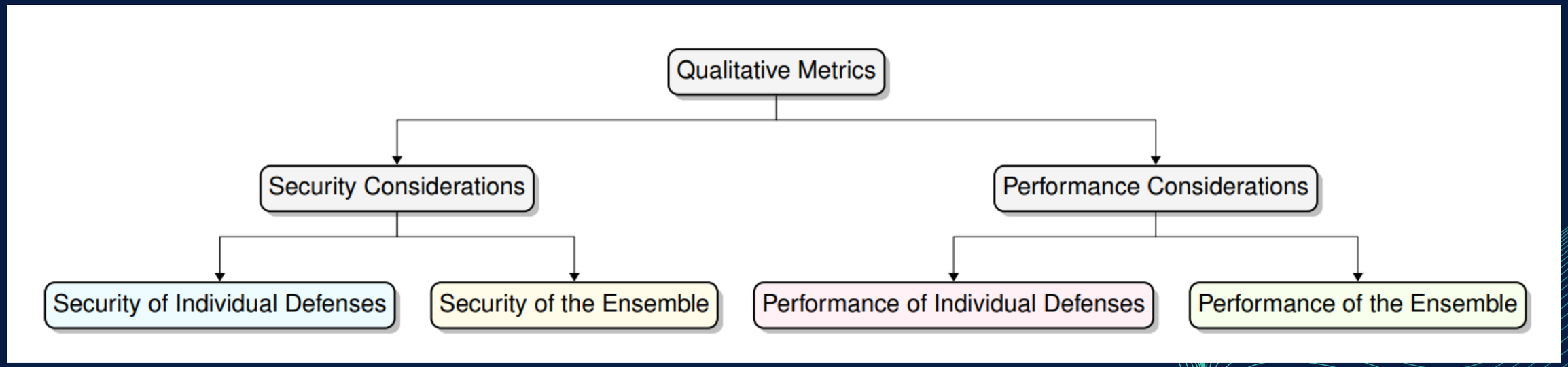
Evaluating Effectiveness of Moving Targets

- Qualitative measures
  1) Security considerations
  2) Performance considerations

- Quantitative measures
  1) Security metrics
  2) Usability metrics

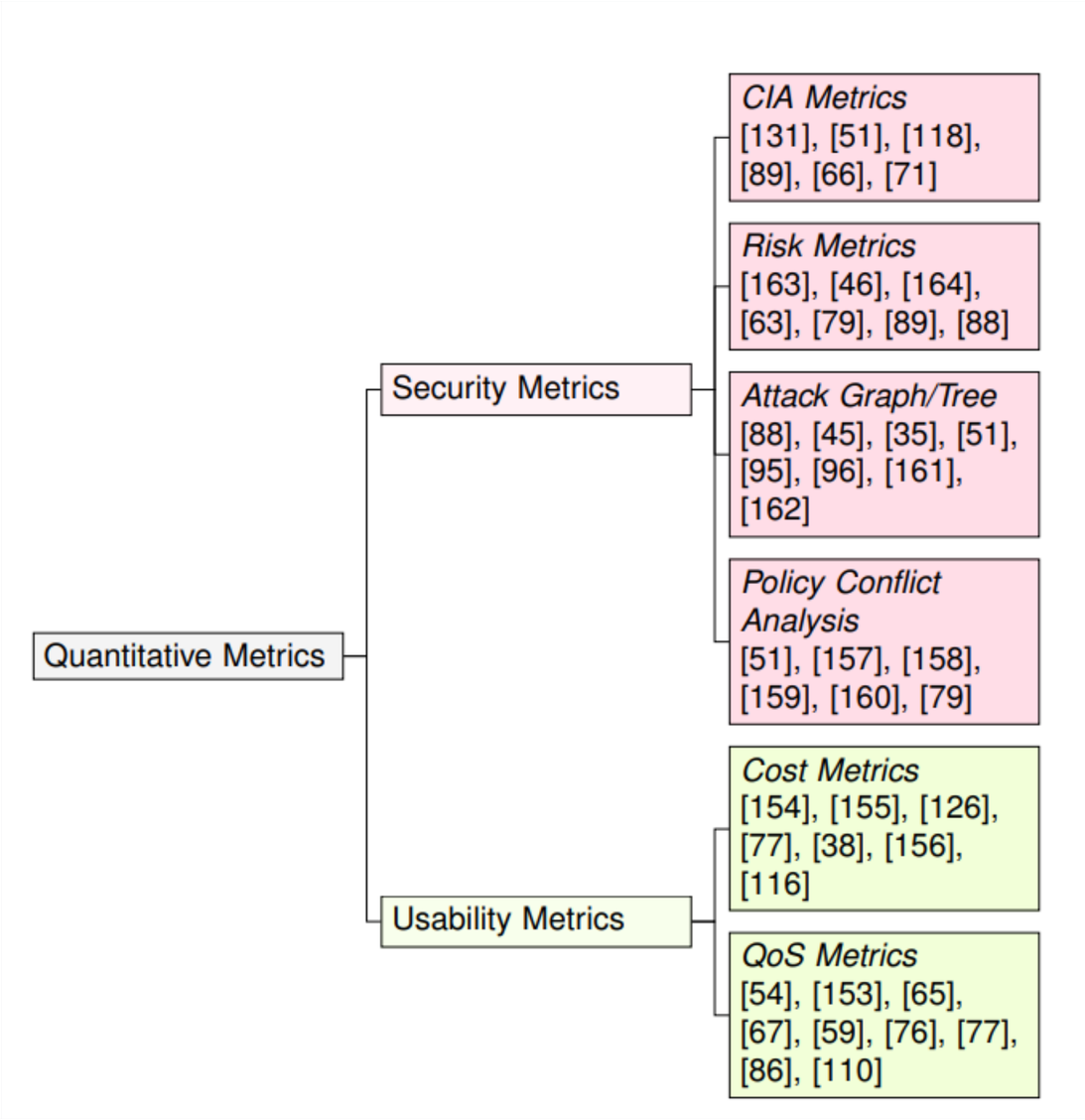
Qualitative measures :

1) Security considerations
2)Performance considerations



Quantitative measures

1) Security metrics
2) Usability metrics



Further Research Opportunities

1) Configuration Set (What to Move)
2) Timing Function (When to Move)
3) Movement Function (How to Move)
4) Qualitative and Quantitaive evaluation
5) Proposing novel MTD’s

ThankYou!