DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

T.E/SEM VI/CBCGS/AIML
Academic Year: 2022-23

NAME	**SINGH SUDHAM DHARMENDRA**
BRANCH	**CSE-(AI&ML)**
ROLL NO.	**57**
SUBJECT	**CRYPTOGRAPHIC AND SYSTEM SECURITY LAB**
COURSE CODE	**CSL602**
PRACTICAL NO.	
DOP	
DOS	

Experiment No. 01

Aim - To Design and Implementation of a product cipher using Substitution and Transposition ciphers.

Theory -

Substitution cipher is a method of encryption by which units of plaintext are replaced with ciphertext according to a regular system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution.

Transposition cipher is a method of encryption by which the positions held by units of plaintext (which are commonly characters or groups of characters) are shifted according to a regular system, so that the ciphertext constitutes a permutation of the plaintext. That is, the order of the units are changed.

Substitution ciphers can be compared with Transposition ciphers. In a transposition cipher, the units of the plaintext are rearranged in a different and usually quite complex order, but the units themselves are left unchanged. By contrast, in a substitution cipher, the units of the plaintext are retained in the same sequence in the ciphertext, but the units themselves are altered.

1. **Caesar Cipher:** In cryptography, a Caesar cipher, also known as a Caesar's cipher, the shift cipher, Caesar's code or Caesar shift, is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a shift of 3, A would be replaced by D, B would become E, and so on. The method is named afterJulius Caesar, who used it to communicate with his generals.

   Example:
   The transformation can be represented by aligning two alphabets; the cipher alphabet is the plain alphabet rotated left or right by some number of positions. For instance, here is a Caesar cipher using a left rotation of three places (the shift parameter, here 3, is used as the key):
   Plain: ABCDEFGHIJKLMNOPQRSTUVWXYZ
   Cipher: DEFGHIJKLMNOPQRSTUVWXYZABC

   When encrypting, a person looks up each letter of the message in the "plain" line and writes down the corresponding letter in the "cipher" line. Deciphering is done in reverse.
   Plaintext: the quick brown fox jumps over the lazy dog
   Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

2. **Columnar Transposition:** In a columnar transposition, the message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order. Both the width of the rows and the permutation of the columns are usually defined by a keyword. For example, the word ZEBRAS is of length 6 (so the rows are of length

6), and the permutation is defined by the alphabetical order of the letters in the keyword. In this case, the order would be "6 3 2 4 1 5".

In a regular columnar transposition cipher, any spare spaces are filled with nulls; in an irregular columnar transposition cipher, the spaces are left blank. Finally, the message is read off in columns, in the order specified by the keyword. For example, suppose we use the keyword ZEBRAS and the message WE ARE DISCOVERED. FLEE AT ONCE. In a regular columnar transposition, we write this into the grid as:
6 3 2 4 1 5 W E A R E D I S C O V E R E D F L E E A T O N C E Q K J E U

The ciphertext is then read off as:
EVLNE ACDTK ESEAQ ROFOJ DEECU WIREE

Procedure / Program -

1. Substitution Encryption
    a. Accept plaintext, P from user
    b. Accept key, K from user.
    c. Generate ciphertext, C=(P+K)mod 26
    d. Display plaintext, P and ciphertext, C.

2. Transposition Encryption:
    a. Count how many letters are in your ciphertext, C (for example, 75) and factor that number (75 =5*5*3).
    b. Create all of the possible matrices to fit this plaintext (in our case, 3x25, 5x15, 15x5,25x3).
    c. Write the ciphertext, C row-wise into these matrices.
    d. Permute the columns. (Shuffle the columns)
    e. Read the matrix in column-wise to get the new ciphertext, C1. f. Display plaintext C and ciphertext C1.

3. Transposition Decryption:
    a. Write the ciphertext, C1 column-by-column.
    b. Permute the columns.
    c. Read the matrix row-wise to recover text C.
    d. Display plaintext C and ciphertext C1

4. Substitution decryption:
    a. Generate plaintext, P=(C-K) mod 26
    b. Display plaintext, P and ciphertext, C.

Conclusion -

A product cipher is a composite of two or more elementary ciphers with the goal of producing a cipher which is more secure than any of the individual components. In product cipher substitution and transposition are applied to create confusion and diffusion in the text message.

Experiment No. 02

Aim - To Implement and analyze RSA cryptosystem and Digital signature scheme using RSA/El Gamal

Theory -

RSA Cryptosystem

This cryptosystem is one the initial systems. It remains the most employed cryptosystem even today. The system was invented by three scholars Ron Rivest, Adi Shamir, and Len Adleman and hence, it is termed as RSA cryptosystem. We will see two aspects of the RSA cryptosystem, firstly generation of key pair and secondly encryption-decryption algorithms.

Generation of RSA Key Pair Each person or a party who desires to participate in communication using encryption needs to generate a pair of keys, namely public key and private key. The process followed in the generation of keys is described below −

- Generate the RSA modulus (n)
  1. Select two large primes, p and q.
  2. Calculate n=p*q. For strong unbreakable encryption, let n be a large number, typically a minimum of 512 bits.

- Find Derived Number (e)
  1. Number e must be greater than 1 and less than (p − 1)(q − 1).
  2. There must be no common factor for e and (p − 1)(q − 1) except for 1. In other words two numbers e and (p – 1)(q – 1) are coprime.

- Form the public key
  1. The pair of numbers (n, e) form the RSA public key and are made public.
  2. Interestingly, though n is part of the public key, difficulty in factoring a large prime number ensures that an attacker cannot find in finite time the two primes (p & q) used to obtain n. This is strength of RSA

- Generate the private key
  1. Private Key d is calculated from p, q, and e. For given n and e, there is a unique number d.
  2. Number d is the inverse of e modulo (p - 1)(q – 1). This means that d is the number less than (p - 1)(q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1)(q - 1).
  3. This relationship is written mathematically as follows − ed = 1 mod (p − 1)(q − 1)

The Extended Euclidean Algorithm takes p, q, and e as input and gives d as output.

Example:

An example of generating an RSA Key pair is given below. (For ease of understanding, the primes p & q taken here are small values. Practically, these values are very high).

- Let two primes be p = 7 and q = 13. Thus, modulus n = pq = 7 x 13 = 91.

- Select e = 5, which is a valid choice since there is no number that is a common factor of 5 and (p − 1)(q − 1) = 6 × 12 = 72, except for 1.
- The pair of numbers (n, e) = (91, 5) forms the public key and can be made available to anyone whom we wish to be able to send us encrypted messages.
- Input p = 7, q = 13, and e = 5 to the Extended Euclidean Algorithm. The output will be d = 29.
- Check that the d calculated is correct by computing − de = 29 × 5 = 145 = 1 mod 72
- Hence, the public key is (91, 5) and the private key is (91, 29).

Encryption and Decryption

Once the key pair has been generated, the process of encryption and decryption are relatively straightforward and computationally easy.

RSA Encryption

- Suppose the sender wishes to send some text message to someone whose public key is (n, e).
- The sender then represents the plaintext as a series of numbers less than n.
- To encrypt the first plaintext P, which is a number modulo n. The encryption process is simple mathematical step as − C = P^e mod n
- In other words, the ciphertext C is equal to the plaintext P multiplied by itself e times and then reduced modulo n. This means that C is also a number less than n.
- Returning to our Key Generation example with plaintext P = 10, we get ciphertext C − C = 105 mod 91

RSA Decryption

- The decryption process for RSA is also very straightforward. Suppose that the receiver of a public-key pair (n, e) has received a ciphertext C.
- Receiver raises C to the power of his private key d. The result modulo n will be the plaintext P. Plaintext = C^d mod n
- Returning again to our numerical example, the ciphertext C = 82 would get decrypted to number 10 using private key 29 − Plaintext = 82^29 mod 91 = 10

Key Pair *Public key: $n = 55, e = 3$* *Private key: $n = 55, d = 7$*			***Key Pair Generation*** *Primes: $p = 5, q = 11$* *Modulus: $n = pq = 55$* *Public exponent: $e = 3$* *Private exponent: $d = 3^{-1} \bmod 20 = 7$*			
Message	***Encryption*** $c = m^3 \bmod n$		***Decryption*** $m = c^7 \bmod n$			
m	$m^2 \bmod n$	$m^3 \bmod n$	$c^2 \bmod n$	$c^3 \bmod n$	$c^6 \bmod n$	$c^7 \bmod n$
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	4	8	9	17	14	2
3	9	27	14	48	49	3
4	16	9	26	14	31	4
5	25	15	5	20	15	5
6	36	51	16	46	26	6
7	49	13	4	52	9	7
8	9	17	14	18	49	8
9	26	14	31	49	36	9

RSA Digital Signature:

A digital signature is a mathematical scheme for demonstrating the authenticity of a digital message or documents. A valid digital signature gives a recipient reason to believe that the message was created by a known sender, that the sender cannot deny having sent the message (authentication and non-repudiation), and that the message was not altered in transit.

To sign: use a private signing algorithm

To verify: use a public verification algorithm

Alice wants to sign a message m. She computes the signature of m (let's call it y) and sends the signed message (m,y) to Bob. Bob gets (m,y), runs the verification algorithm on it. The algorithm returns "true" iff y is Alice's signature of m.

The basic protocol:

1. Alice encrypts the document with her private key.
2. Alice sends the signed document to Bob.
3. Bob decrypts the document with Alice's public key.

Procedure/ Program -

1. Take choice as an input
2. If choice=1
    a. Choose two large prime numbers P and Q.
    b. Calculate $N = P \times Q$.
    c. Select the public key (i.e. the encryption key) E such that it is not a factor of (P-1) and (Q-1).
    d. Select the private key (i.e. the decryption key) E such that the following equation is true:
    $(D \times E) \bmod (P-1) \times (Q-1) = 1$
    e. For encryption, calculate the ciphertext CT from the plain text PT as follows:
    $CT = PT^E \bmod N$
    f. Send CT as the cipher text to the receiver.
    g. For decryption, calculate the plaintext PT from the ciphertext CT as follows:
    $PT = CT^D \bmod N$.
3. If choice=2
    a. Alice chooses secret odd primes p,q and computes n=pq.
    b. Alice chooses e_A with $\gcd(e_A, \Phi(n))=1$.
    c. Alice computes $d_A = e_A^{-1} \bmod \Phi(n)$.
    d. Alice's signature is $y = m^d{}_A \bmod n$.
    e. The signed message is (m,y).
    f. Bob can verify the signature by calculating $z = y^e{}_a \bmod n$. (The signature is valid iff m=z).

Conclusion -

RSA is a strong encryption algorithm. RSA implements a public-key cryptosystem that allows secure communications and "digital signatures", and its security rests in part on the difficulty of factoring large numbers.

Experiment No. 03

Aim - To Implement Diffie Hellman Key exchange algorithm

Theory -

The Diffie-Hellman Algorithm

Diffie–Hellman key exchange (D–H) is a specific method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols as originally conceptualized by Ralph Merkle and named after Whitfield Diffie and Martin Hellman. D–H is one of the earliest practical examples of public key exchange implemented within the field of cryptography. Traditionally, secure encrypted communication between two parties required that they first exchange keys by some secure physical channel, such as paper key lists transported by a trusted courier. The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.

The Diffie–Hellman key exchange algorithm solves the following dilemma. Alice and Bob want to share a secret key for use in a symmetric cipher, but their only means of communication is insecure. Every piece of information that they exchange is observed by their adversary Eve. How is it possible for Alice and Bob to share a key without making it available to Eve? At first glance it appears that Alice and Bob face an impossible task. It was a brilliant insight of Diffie and Hellman that the difficulty of the discrete logarithm problem for F*pprovides a possible solution.

The first step is for Alice and Bob to agree on a large prime p and a nonzero integer g modulo p. Alice and Bob make the values of p and g public knowledge; for example, they might post the values on their web sites, so Eve knows them, too. For various reasons to be discussed later, it is best if they choose g such that its order in F* is a large prime.

The next step is for Alice to pick a secret integer a that she does not reveal to anyone, while at the same time Bob picks an integer b that he keeps secret. Bob and Alice use their secret integers to compute

$$\underbrace{A \equiv g^a \pmod{p}}_{\text{Alice computes this}} \quad \text{and} \quad \underbrace{B \equiv g^b \pmod{p}}_{\text{Bob computes this}}.$$

They next exchange these computed values, Alice sends A to Bob and Bob sends B to Alice. Note that Eve gets to see the values of A and B, since they are sent over the insecure communication channel.

Finally, Bob and Alice again use their secret integers to compute

$$\underbrace{A' \equiv B^a \pmod{p}}_{\text{Alice computes this}} \quad \text{and} \quad \underbrace{B' \equiv A^b \pmod{p}}_{\text{Bob computes this}}.$$

The values that they compute, A0 and B0 respectively, are actually the same, since

$$A' \equiv B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \equiv B' \pmod{p}.$$

This common value is their exchanged key. The Diffie-Hellman key exchange algorithm is summarized in Table

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p and an integer g having large prime order in F^*.	
Private Computations	
Alice	**Bob**
Choose a secret integer. Compute $A \equiv g^a \pmod{p}$.	Choose a secret integer. Compute $B \equiv g^b \pmod{p}$.
Public Exchange of Values	
Alice sends A to Bob ⟶	
AB ⟵ Bob sends B to Alice	
Further Private Computations	
Alice	**Bob**
Compute the number $B^a \pmod{p}$.	Compute the number $A^b \pmod{p}$.
The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.	

Table 1. Diffie-Hellman Key Exchange

Procedure / Program -

i. Firstly, Alice and Bob agree on two large prim numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.

ii. Alice chooses another large random number x, and calculates A such that:

$$A = g^x \bmod n$$

iii. Alice sends the number A to Bob.

iv. Bob independently chooses another large random integer y and calculates B such that:

$$B = g^y \bmod n$$

v. Bob sends the number B to Alice.

vi. A now computes the secret key K_1 as follows:

$$K_1 = B^x \bmod n$$

vii. B now computes the secret key K_2 as follows:

$$K_2 = A^y \bmod n$$

Conclusion -

The Diffie-Hellman key exchange algorithm is used to make a secure channel to share secret keys between sender and receiver. But man in the middle attack is possible on this algorithm as values of n and g are publically known.

Experiment No. 04

Aim - For varying message sizes, test integrity of message using MD-5, SHA-1, and analyze the performance of the two protocols. Use crypt APIs.

Theory -

MD5 (Message Digest algorithm 5) is a widely used cryptographic hash function with a 128- bit hash value. An MD5 hash is typically expressed as a 32-digit hexadecimal number. MD5 processes a variable length message into a fixed length output of 128 bits. The input message is broken up into chunks of 512-bit blocks (sixteen 32 bit little endian integers) ; The message is padded so that its length is divisible by 512. The padding works as follows: first a single bit, 1, is appended to the end of the message. This is followed by as many zeros as are required to bring the length of the message up to 64 bits less than a multiple of 512. The remaining bits are filled up with a 64bit integer representing the length of the original message, in bits.

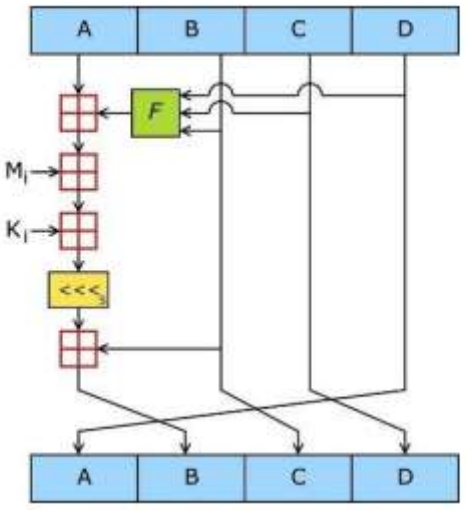


Figure 1: One MD5 operation

MD5 consists of 64 of these operations, grouped in four rounds of 16 operations. F is a nonlinear function; one function is used in each round. Mi denotes a 32bit block of the message input, and Ki denotes a 32bit constant, different for each operation.

The main MD5 algorithm operates on a 128bit state, divided into four 32bit words, denoted A, B, C and D. These are initialized to certain fixed constants. The main algorithm then operates on each 512bit message block in turn, each block modifying the state. The processing of a message block consists of four similar stages, termed rounds; each round is composed of 16 similar operations based on a nonlinear function F, modular addition, and left rotation.

Figure 1 illustrates one operation within a round. There are four possible functions F; a different one is used in each round:

$$F(X,Y,Z) = (X \wedge Y) \vee (\neg X \wedge Z)$$
$$G(X,Y,Z) = (X \wedge Z) \vee (Y \wedge \neg Z)$$
$$I(X,Y,Z) = Y \oplus (X \vee \neg Z)$$

$\oplus, \wedge, \vee, \neg$ denote the XOR, AND, OR and NOT operations respectively.

Procedure / Program -

1. Append Padding Bits

The message is "padded" (extended) so that its length (in bits) is congruent to 448, modulo 512. That is, the message is extended so that it is just 64 bits shy of being a multiple of 512 bits long. Padding is always performed, even if the length of the message is already congruent to 448, modulo 512. Padding is performed as follows: a single "1" bitis appended to the message, and then "0" bits are appended so that the length in bits ofthe padded message becomes congruent to 448, modulo 512. In all, at least one bit andat most 512 bits are appended.

2. Append Length

A 64-bit representation of b (the length of the message before the padding bits were added) is appended to the result of the previous step. In the unlikely event that b is greater than 2^64, then only the low order 64 bits of b are used. (These bits are appended as two 32bit words and appended low order word first in accordance withthe previous conventions.) At this point the resulting message (after padding with bits and with b) has a length that is an exact multiple of 512 bits. Equivalently, this message has a length that is an exact multiple of 16 (32 bit) words. Let M[0 ... N1] denote the words of the resulting message, where N is a multiple of 16.

3. Initialize MD Buffer

A fourword buffer (A,B,C,D) is used to compute the message digest. Here each of A, B, C, D is a 32bit register. These registers are initialized to the following values in hexadecimal, loworder bytes first):

4. Process Message in 16Word Blocks

We first define four auxiliary functions that each take as input three 32bit words and produce as output one 32bit word.

Conclusion -

The main aim of the message digest algorithm is to ensure integrity of the message. The strength of the MD5 algorithm lies in the chaining function, because of which integrity of message cannot be compromised.

Experiment No. 05

Aim - To explore wireless security tools like Kismet, NetStumbler etc.

Theory - Wireless Security Tools

1. **Kismet**

Essential tools

- kismet – packet sniffer
- Spectrum analyzers: airview, wispy
- Netstumbler (windows)
- etherape (not a strong admin tool – just a quick visual overview)
- General networking tools: wireshark, ntop, mrtg, rrdtool, nmap
- WEP/WPA/WPA2 cracking: aircrack etc

What is kismet?

- Kismet is an 802.11 layer2 wireless network detector, sniffer, and intrusion detection system.
- Works in raw monitoring (rfmon) mode, and (with appropriate hardware) can sniff 802.11b, 802.11a, 802.11g, and 802.11n traffic.
- It is passively collecting packets and detecting standard named networks, detecting (and given time, decloaking) hidden networks, and presence of non beaconing networks via data traffic.

Strengths

Server – Client architecture

Drones: distributed kismet servers running on remote devices, reporting back to central server, allow for the building of distributed reporting and intrusion detection systems

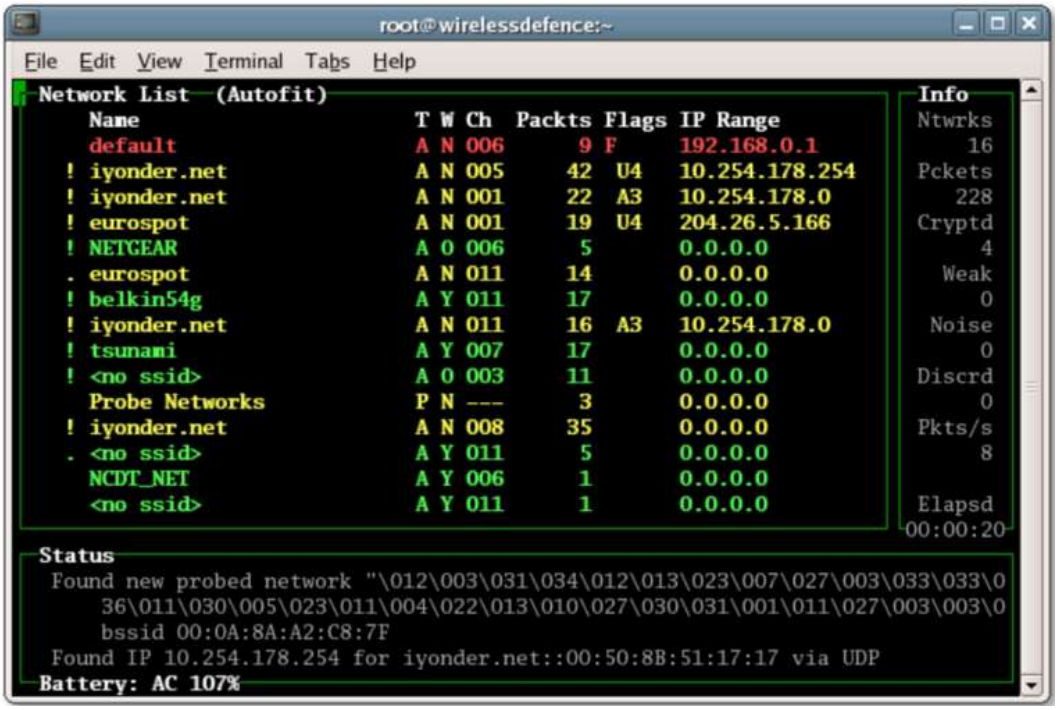
Kismet is powerful - especially when combined with other tools like wireshark, nmap

Installing I

- The following guide assumes you are on Ubuntu 9.10 / GNU/Linux - but works for other systems accordingly.
- Get kismet via apt-get (or synaptic) $ aptget install kismet
- edit /etc/kismet.conf - Definition of sources is a must. Sources are defined as: source=sourcetype,interface,name[,initialchannel] For the list of sourcetypes, see the README or online documentation.

Installing II

- $ vi /ect/kismet.conf source=sourcetype,interface,name[,initialchannel] e.g. source=ipw3945,wlan0,my_internal_card
- start kismet $ kismet
  Start screen



What does Kismet show?

- List of SSIDs Note: it also shows networks with hidden SSIDs / no beacons - just blank! If a client associates with those, you will also see the SSID.

What does Kismet show?

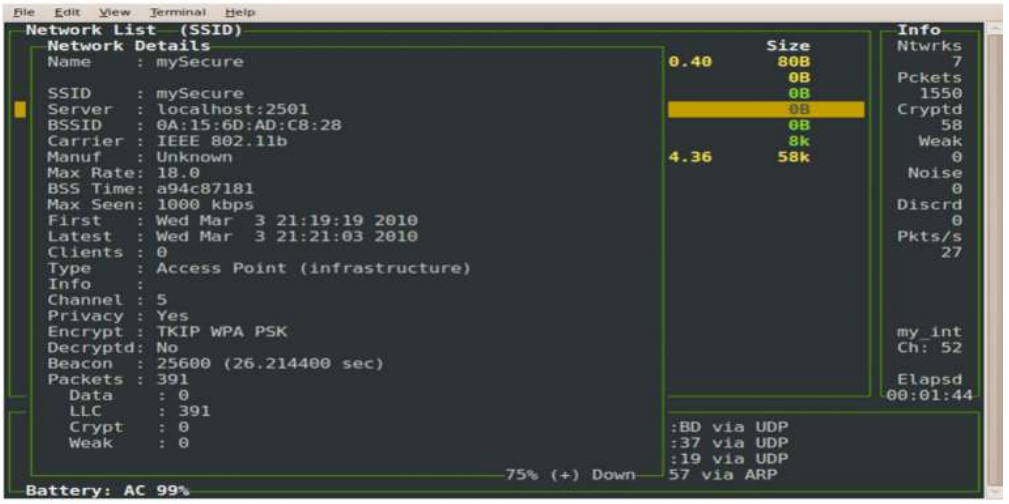
- T = Type P Probe request - no associated connection yet
- A Access point - standard wireless network
- H Ad-hoc - point to point wireless network
- T Turbocell - Turbocell aka Karlnet or Lucent Router
- G Group - Group of wireless networks
- D Data - Data only network with no control packets

What does Kismet show?

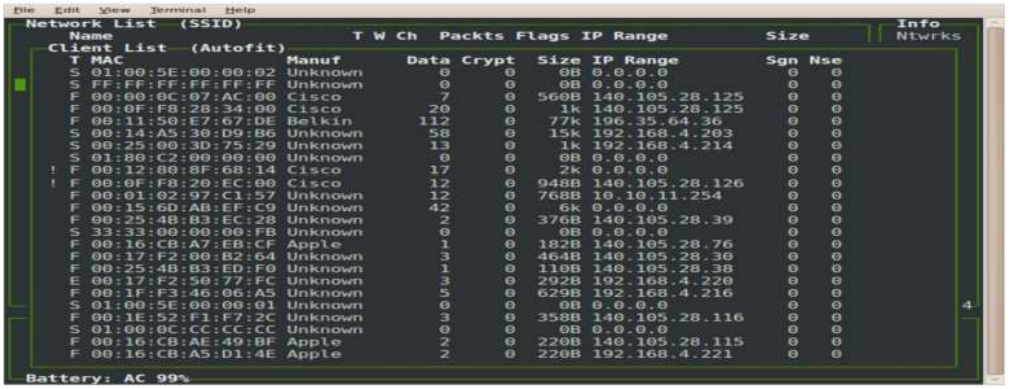
- W = Encryption
- Colour = Network/Client Type: Yellow Unencrypted Network Red Factory default settings in use! Green Secure Networks (WEP, WPA etc..) Blue SSID cloaking on / Broadcast SSID disabled Options
- (Some of the) Options:

➔ c Show clients in current network
➔ h Help
➔ i Detailed info about current network
➔ s Sort network list
➔ r Packet rate graph
➔ a Statistics
➔ p Dump packet type
➔ Q Quit

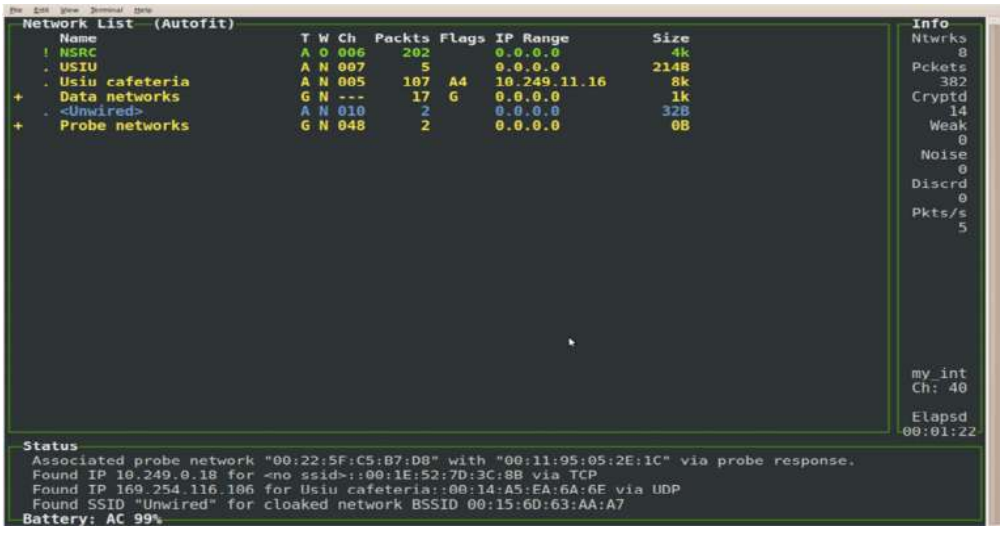
Network info



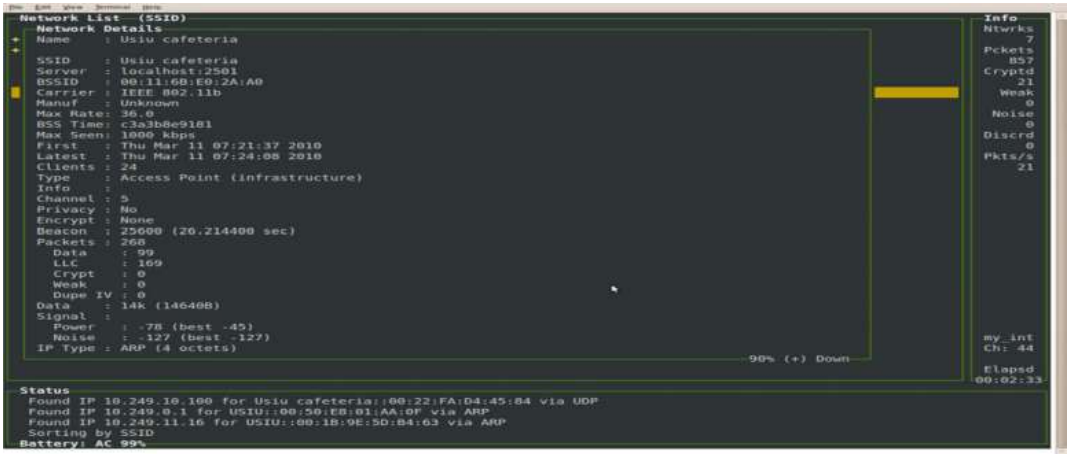
Client info



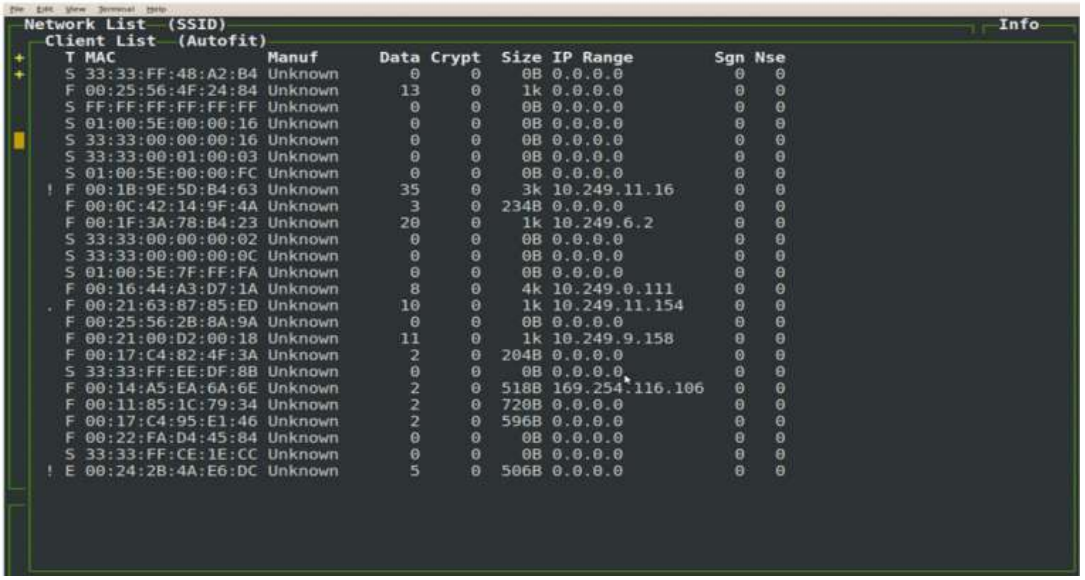
Kismet scan USIU



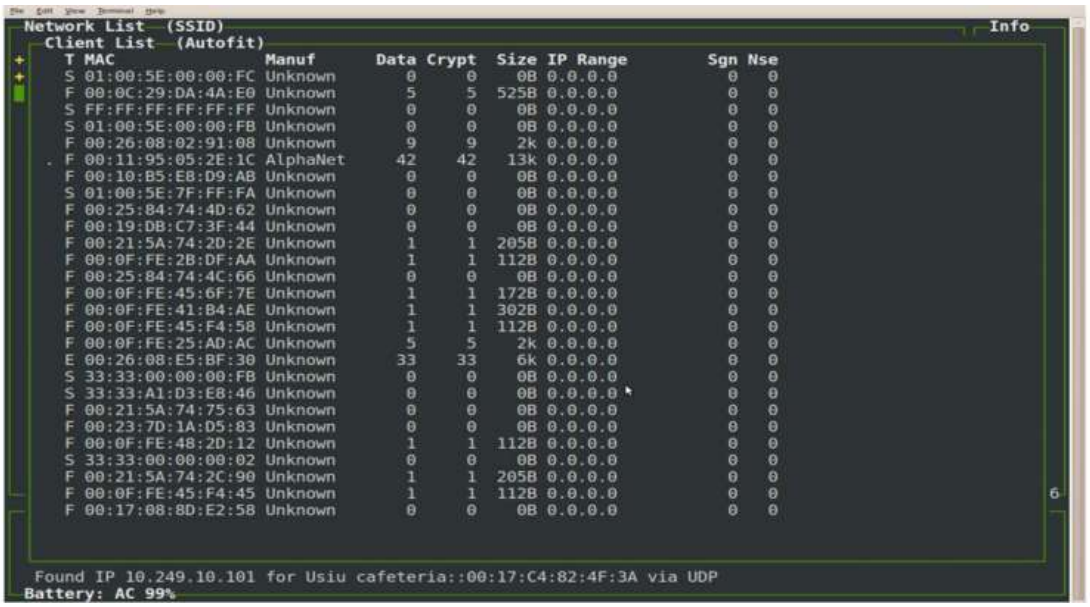
Kismet scan USIU



Kismet scan USIU



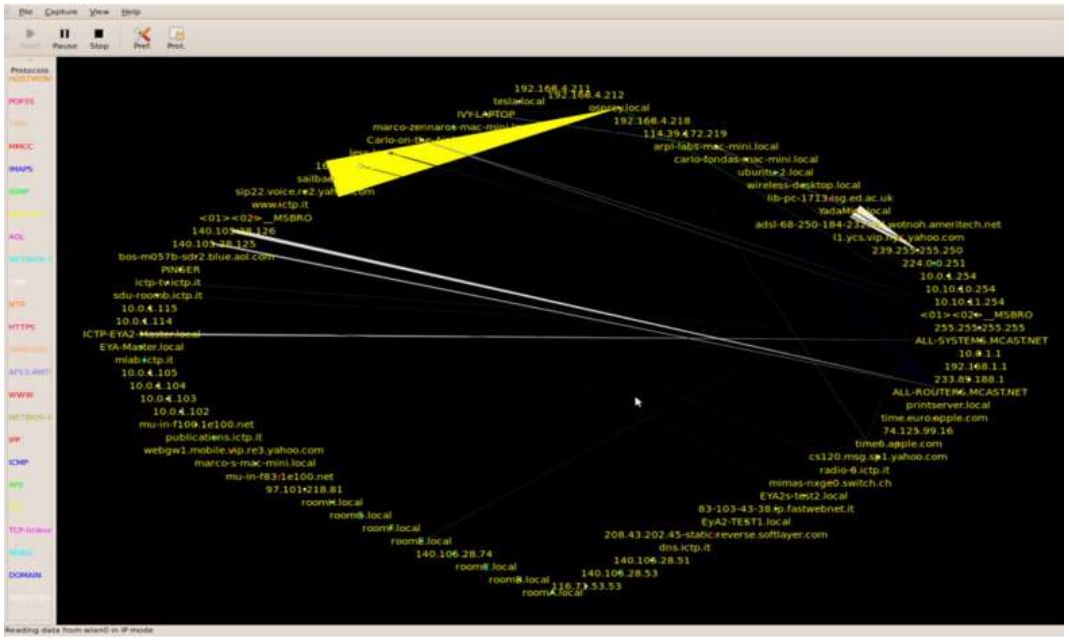
Kismet scan USIU



What is Etherape?

- Etherape is not really a security tool, but it gives a very useful quick first view of traffic in your network.
- For example, in case you have a spam virus in your network, you will see this immediately.
- It also gives you a good feel for what various applications, such as skype or torrent clients, are doing to your network.

Etherape screenshot



2. NetStumbler

For a Swiss Army knife of wireless network diagnostics, “NetStumbler” is saddled with a somewhat unfortunate name. Although it implies a sort of blind luck, NetStumbler is actually most useful for pinpointing details of a wireless network, helping you configure, secure, optimize and discover.

NetStumbler calls itself “beggarware,” meaning that it is free (but not open source), although they request a $50 donation from commercial and government users. The latest version (0.4.0 as of this writing) is available for download from netstumbler.com and stumbler dot net.

- The Right Hardware for the Job

Requiring Windows 2000, XP or newer, NetStumbler functions best with a supported wireless card. Determining precisely which cards are fully supported can take some sleuthing.

NetStumbler fully supports cards based on the Proxim 8410-WD and 8420-WD, which have most commonly been sold under the names Orinoco Classic Gold and Orinoco Gold. Other cards based around this chipset include the Dell TrueMobile 1150, Compaq WL110, and Avaya Wireless 802.11b PC Card. Also supported are cards based on the Intersil (now owned by Conexant) Prism and Prism2 wireless chipsets, such as the popular D-Link DWL-650.

Unfortunately, there is no single comprehensive source of information on wireless card chipsets and retail models. Seattle Wireless maintains a wiki, and NetStumbler hosts user-submitted compatibility reports, although they do not indicate which chipset a card uses.
Wireless cards which are fully supported in NetStumbler are able to report accurate noise and signal strength levels. The latest 0.4 version of NetStumbler partially supports most wireless cards, but those without full support will not be reliable for noise and strength readings, and may cause instability in NetStumbler itself.

- Getting Off the Ground

NetStumbler and Windows Wireless Zero Configuration service do not play well together. The 0.4 version of NetStumbler includes a feature called “Auto Reconfigure” which you can enable by clicking the “two gears” icon on the toolbar or through the View, Options menu. With Auto Reconfigure enabled, NetStumbler will make an effort to stop the WZC service upon launching, and restore it upon exit. Alternatively, you can take control of the situation by enabling and disabling WZC yourself (Windows Control Panel, Administrative Tools, Services, Wireless Zero Configuration).

Some wireless cards will not see all available access points unless their SSID is set to blank or “ANY.” Again, NetStumbler with Auto Reconfigure enabled will attempt to set your wireless card accordingly while running.

When NetStumbler launches, you may see two entries for your wireless card under the Device menu. The first entry includes the chipset name for your card (such as “Prism2”), whereas the second reads “NDIS.”

Which to use? The easiest way to tell is to run NetStumbler within reach of a known functioning access point. Choose the first device entry and see if the AP shows up in NetStumbler’ window.
If yes, your card is fully supported by NetStumbler. Do not use the NDIS device.
If the first device entry does not detect the AP in a few seconds, try the NDIS entry. If this works, your card is partially supported, and will not return reliable data for noise and signal strength. You can continue to use NetStumbler’s other features in NDIS mode. If neither device driver detects the AP, try using your wireless card’s management utility tomanually set its SSID to blank or “ANY”. If none of these combinations detect the AP under NetStumbler, you may have a funky wireless card which cannot be used with NetStumbler.

- Finding Access Points

While running NetStumbler, the right-hand pane shows APs currently detected and available under the current view filter. By default, you have no view filter set, so all detected APs are displayed.

Each AP listing is marked with a colored dot indicating the signal strength to that access point, alongside its MAC address, the unique identifier assigned to each network device. The colors range from red (signal too low) to yellow (marginal) to green (good). A grey dot marks an AP which had been detected but is now gone. A lock appears on the dot icon when the AP is operating with encryption enabled.

For many NetStumbler users, detecting available APs is the software's primary feature. Typically, the software is run on a mobile computer, which you either carry to some location or drive around within the car, scanning the air for detected access points. The practice of hunting for access points has come to be known as "war-driving," another unfortunate term, since detecting APs alone is not itself an aggressive or malicious act.

To clear the record, NetStumbler does not connect you to available access points. While NetStumbler can detect them, you still need to rely on either Windows or your wireless card's management software to join a wireless network. Since your connection software also displays available networks, you may wonder, why bother with NetStumbler?

NetStumbler may better disambiguate access points which share an SSID, for one example. But more often, NetStumbler can continuously scan for access points as you roam about an area, presenting a convenient log of its activity, including audio notification. This functionality is typically not available from Windows' or vendor-provided wireless client software.

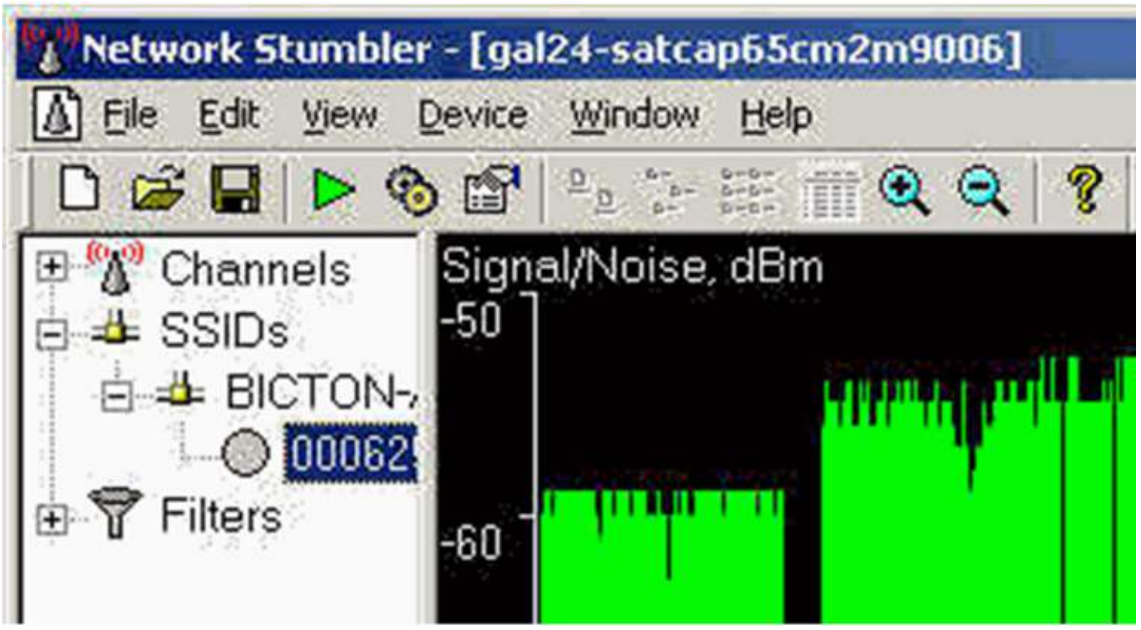
- Exploring Access Points

The left pane of NetStumbler is an Explorer-like interface for navigating available wireless access points. Under the "Channels" heading, you will find all detected access points listed under their channel frequencies. Under "SSIDs," you will find all detected access points sorted by their network name. You may find two or more APs listed under the same SSID. This could indicate two separate wireless networks overlapping in range, which could cause problems for clients.

Alternatively, it may indicate one wireless network with multiple APs available from your current location.

In cases where you find multiple APs sharing the same SSID, look at the "Subnet" field in the right pane. Here you will see which IP network the APs are operating on.

Signal-to-Noise Graphs



NetstumblerClicking on an AP's MAC address in the left pane will replace the right pane with a live signal-to-noise graph. Note that this graph is accurate only if your network card is fully supported by NetStumbler. Signal-to-noise readings can be a powerful tool for troubleshooting your network and optimizing AP or antenna placement.

The graph overlays two sets of values – signal strength (green) and noise (red), measured in dBm. The "taller" your green plot, the stronger your signal; likewise, the taller your red plot, the more noise is present. For the best wireless performance, you want to maximize your signal and minimize your noise. Typical sources of noise in the Wi-Fi 2.4GHz range include microwave ovens, cordless phones, wireless video transmitters, and perhaps neighboring wireles networks.

You can also observe the consistency of your graph to determine the presence of sources of intermittent interference. Partially supported network cards will produce signal strength (green) plots which may or may not be accurate, along with no noise (red) plots.

- Access Point Filters

The "Filters" item in the left pane expands to a list of criteria for filtering the right pane list of available access points. If you click the "Encryption Off" filter, only open APs will be listed on the right. Some of the filters are quite technical, and are only useful in specialized situations.
One thing to keep in mind – if you're not seeing an AP on the right that you know is available, check that you have not selected a filter which may exclude it from appearing.

- Mobile Tracking with GPS

If your NetStumbling PC sports an attached GPS receiver, you can enable GPS support in NetStumbler to track the location of detected APs. Use the View, Options, GPS menu to configure your receiver. NetStumbler will fill in the latitude and longitude fields in the right pane, and will record GPS data in logs that you can output through the File, Export menu.

- Extending NetStumbler

NetStumbler exposes a small library of functions which can be accessed through active scripting languages under Windows, including VBScript, JScript, and ActiveState's PerlScript and Python. You can connect NetStumbler to external scripts through the View, Options, Scripting menu.
One popular approach to scripting connects NetStumbler events to text-to-speech output, particularly valuable for so-called "war-driving." More details are available in the NetStumbler support forum.

- Further Support

NetStumbler is supported through its online community. There are FAQs and newbie forums that veteran NetStumblers would strongly prefer you read. For whatever reason, NetStumblers are not the most welcoming of online communities, often handling newcomers' questions with short, world-weary replies. But the support is free, and you don't have to be roommates with them.

Conclusion -

We had successfully explored wireless security tools like Kismet, NetStumbler etc.

Experiment No. 06

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

Theory -

1. **WHOIS :** WHOIS is the Linux utility for searching an object in a WHOIS database. The WHOIS database of a domain is the publicly displayed information about a domain's ownership, billing, technical, administrative, and nameserver information. Running a WHOIS on your domain will look the domain up at the registrar for the domain information. All domains have WHOIS information. WHOIS database can be queried to obtain the following information via WHOIS:

    - Administrative contact details, including names, email addresses, and telephone numbers
    - Mailing addresses for office locations relating to the target organization
    - Details of authoritative name servers for each given domain

    Example: **Querying Facebook.com**

```
ssc@ssc-OptiPlex-380:~$ whois facebook.com
Whois Server Version 2.0
Domain names in the .com and .net domains can now be registered
with many different competing registrars. Go to http://www.internic.net

for detailed information.
Server Name: FACEBOOK.COM.BRETLANDTRUSTMERCHANDISINGDEPART.COM

IP Address: 69.63.176.11
Registrar: GOOGLE INC.
Whois Server: whois.rrpproxy.net
Referral URL: http://domains.google.com
Server Name:
FACEBOOK.COM.DISABLE.YOUR.TIMELINE.NOW.WITH.THE.ORIGINAL.TIMELIN
E-REMOVE.NET
IP Address: 8.8.8.8
Registrar: ENOM, INC.
Whois Server: whois.enom.com
Referral URL: http://www.enom.com
Server Name:
FACEBOOK.COM.GET.ONE.MILLION.DOLLARS.AT.WWW.UNIMUNDI.COM
IP Address: 209.126.190.70
Registrar: PDR LTD. D/B/A PUBLICDOMAINREGISTRY.COM
Whois Server: whois.PublicDomainRegistry.com
Referral URL: http://www.PublicDomainRegistry.com
Server Name: FACEBOOK.COM.LOVED.BY.WWW.SHQIPHOST.COM
```

IP Address: 46.4.210.254
Registrar: ONLINENIC, INC.
Whois Server: whois.onlinenic.com
Referral URL: http://www.OnlineNIC.com
Server Name: FACEBOOK.COM.MORE.INFO.AT.WWW.BEYONDWHOIS.COM
IP Address: 203.36.226.2
Registrar: INSTRA CORPORATION PTY, LTD.
Whois Server: whois.instra.net
Referral URL: http://www.instra.com
Server Name:
FACEBOOK.COM.ZZZZZ.GET.LAID.AT.WWW.SWINGINGCOMMUNITY.COM
IP Address: 69.41.185.229
Registrar: TUCOWS DOMAINS INC.
Whois Server: whois.tucows.com
Referral URL: http://www.tucowsdomains.com
Domain Name: FACEBOOK.COM
Registrar: MARKMONITOR INC.
Sponsoring Registrar IANA ID: 292
Whois Server: whois.markmonitor.com
Referral URL: http://www.markmonitor.com
Name Server: A.NS.FACEBOOK.COM
Name Server: B.NS.FACEBOOK.COM
Status: clientDeleteProhibited http://www.icann.org/epp#clientDeleteProhibited
Status:clientTransferProhibited http://www.icann.org/epp#clientTransferProhibited
Status:clientUpdateProhibited http://www.icann.org/epp#clientUpdateProhibited
Status:serverDeleteProhibited
http://www.icann.org/epp#serverDeleteProhibited
Status:serverTransferProhibited http://www.icann.org/epp#serverTransferProhibited
Status:serverUpdateProhibited http://www.icann.org/epp#serverUpdateProhibited
Updated Date: 28-sep-2012
Creation Date: 29-mar-1997
Expiration Date: 30-mar-2020
>>> Last update of whois database: Fri, 17 Jul 2015 04:12:12 GMT <<<
The Registry database contains ONLY .COM, .NET, .EDU domains and Registrars.
For more information on Whois status codes, please visit
https://www.icann.org/resources/pages/epp-status-codes-2014-06-16-en.
Domain Name: facebook.com
Registry Domain ID: 2320948_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.markmonitor.com
Registrar URL: http://www.markmonitor.com
Updated Date: 2014-10-28T12:38:28-0700
reation Date: 1997-03-28T21:00:00-0800
Registrar Registration Expiration Date: 2020-03-29T21:00:00-0700
Registrar: MarkMonitor, Inc.
Registrar IANA ID: 292
Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
Registrar Abuse Contact Phone: +1.2083895740

Domain Status: clientUpdateProhibited (https://www.icann.org/epp#clientUpdateProhibited)
Domain Status: clientTransferProhibited (https://www.icann.org/epp#clientTransferProhibited)
Domain Status: clientDeleteProhibited (https://www.icann.org/epp#clientDeleteProhibited)
Registry Registrant ID:
Registrant Name: Domain Administrator
Registrant Organization: Facebook, Inc.
Registrant Street: 1601 Willow Road,
Registrant City: Menlo Park
Registrant State/Province: CA
Registrant Postal Code: 94025
Registrant Country: US
Registrant Phone: +1.6505434800
Registrant Phone Ext:
Registrant Fax: +1.6505434800
Registrant Fax Ext:
Registrant Email: domain@fb.com
Registry Admin ID:
Admin Name: Domain Administrator
Admin Organization: Facebook, Inc.
Admin Street: 1601 Willow Road,
Admin City: Menlo Park
Admin State/Province: CA
Admin Postal Code: 94025
Admin Country: US
Admin Phone: +1.6505434800
Admin Phone Ext:
Admin Fax: +1.6505434800
Admin Fax Ext:
Admin Email: domain@fb.com
Registry Tech ID:
Tech Name: Domain Administrator
Tech Organization: Facebook, Inc.
Tech Street: 1601 Willow Road,
Tech City: Menlo Park
Tech State/Province: CA
Tech Postal Code: 94025
Tech Country: US
Tech Phone: +1.6505434800
Tech Phone Ext:
Tech Fax: +1.6505434800
Tech Fax Ext:
Tech Email: domain@fb.com
Name Server: b.ns.facebook.com
Name Server: a.ns.facebook.com
DNSSEC: unsigned
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/
>>> Last update of WHOIS database: 2015-07-16T21:08:30-0700 <<<

The Data in MarkMonitor.com's WHOIS database is provided by MarkMonitor.com for information purposes, and to assist persons in obtaining information about or related to a domain name registration record. MarkMonitor.com does not guarantee its accuracy. By submitting a WHOIS query, you agree that you will use this Data only for lawful purposes and that, under no circumstances will you use this Data to:

1) allow, enable, or otherwise support the transmission of mass unsolicited, commercial advertising or solicitations via e-mail (spam); or
2) enable high volume, automated, electronic processes that apply to MarkMonitor.com (or its systems).

MarkMonitor.com reserves the right to modify these terms at any time.
By submitting this query, you agree to abide by this policy.
MarkMonitor is the Global Leader in Online Brand Protection.
MarkMonitor Domain Management(TM)
MarkMonitor Brand Protection(TM)
MarkMonitor AntiPiracy(TM)
MarkMonitor AntiFraud(TM)
Professional and Managed Services
Visit MarkMonitor at http://www.markmonitor.com
Contact us at +1.8007459229
In Europe, at +44.02032062220
ssc@ssc-OptiPlex-380:~$

2. **Dig :** Dig is a networking tool that can query DNS servers for information. It can be very helpful for diagnosing problems with domain pointing and is a good way to verify that your configuration is working. The most basic way to use dig is to specify the domain we wish to query:

Example:

```
$ dig duckduckgo.com
; <<>> DiG 9.8.1-P1 <<>> duckduckgo.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64399
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;duckduckgo.com. IN A
;; ANSWER SECTION:
duckduckgo.com. 99 IN A 107.21.1.61
duckduckgo.com. 99 IN A 184.72.106.253
duckduckgo.com. 99 IN A 184.72.106.52
duckduckgo.com. 99 IN A 184.72.115.86
;; Query time: 33 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Aug 23 14:26:17 2013
;; MSG SIZE rcvd: 96
```

The lines above act as a header for the query performed. It is possible to run dig in batch mode,

so proper labeling of the output is essential to allow for correct analysis.
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 64399
;; flags: qr rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 0, ADDITIONAL: 0
The next section gives us a technical summary of our query results. We can see that the query was successful, certain flags were used, and that 4 "answers" were received.
;; QUESTION SECTION:
;duckduckgo.com. IN A
;; ANSWER SECTION:
duckduckgo.com. 99 IN A 107.21.1.61
duckduckgo.com. 99 IN A 184.72.106.253
duckduckgo.com. 99 IN A 184.72.106.52
duckduckgo.com. 99 IN A 184.72.115.86
The above section of the output contains the actual results we were looking for. It restates the query and then returns the matching DNS records for that domain name.
Here, we can see that there are four "A" records for "duckduckgo.com". By default, "A" records
are returned. This gives us the IP addresses that the domain name resolves to.
The "99" is the TTL (time to live) before the DNS server rechecks the association between the
domain name and the IP address. The "IN" means the class of the record is a standard internet class.
;; Query time: 33 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Aug 23 14:26:17 2013
;; MSG SIZE rcvd: 96

These lines simply provide some statistics about the actual query results. The query time can be indicative of problems with the DNS servers

3. **Traceroute :** traceroute prints the route that packets take to a network host. Traceroute utility uses the TTL field in the IP header to achieve its operation. For users who are new to the TTL field, this field describes how much hops a particular packet will take while traveling on the network. So, this effectively outlines the lifetime of the packet on the network. This field is usually set to 32 or 64. Each time the packet is held on an intermediate router, it decreases the TTL value by 1. When a router finds the TTL value of 1 in a received packet then that packet is not forwarded but instead discarded. After discarding the packet, router sends an ICMP error message of —Time exceeded‖ back to the source from where packet generated. The ICMP packet that is sent back contains the IP address of the router. So now it can be easily understood that traceroute operates by sending packets with TTL value starting from 1 and then incrementing by one each time. Each time a router receives the packet, it checks the TTL field, if TTL field is 1 then it discards the packet and sends the ICMP error packet containing its IP address and this is what traceroute requires. So traceroute incrementally fetches the IP of all the routers between the source and the destination.

Example:
$traceroute example.com
traceroute to example.com (64.13.192.208), 64 hops max, 40 byte packets

1 72.10.62.1 (72.10.62.1) 1.000 ms 0.739 ms 0.702 ms
2 10.101.248.1 (10.101.248.1) 0.683 ms 0.385 ms 0.315 ms
3 10.104.65.161 (10.104.65.161) 0.791 ms 0.703 ms 0.686 ms
4 10.104.65.161 (10.104.65.161) 0.791 ms 0.703 ms 0.686 ms
5 10.0.10.33 (10.0.10.33) 2.652 ms 2.260 ms 5.353 ms
6 acmkokeaig.gs01.gridserver.com (64.13.192.208) 3.384 ms 8.001 ms 2.439 ms

4. **Nslookup :** The nslookup command is used to query internet name servers interactively for information. nslookup, which stands for "name server lookup", is a useful tool for finding out information about a named domain. By default, nslookup will translate a domain name to an IP address (or vice versa). For instance, to find out what the IP address of microsoft.com is, you could run the command:

   Example:
   $nslookup microsoft.com
   Server: 8.8.8.8
   Address: 8.8.8.8#53
   Non-authoritative answer:
   Name: microsoft.com
   Address: 134.170.185.46
   Name: microsoft.com
   Address: 134.170.188.221

   Here, 8.8.8.8 is the address of our system's Domain Name Server. This is the server our system is configured to use to translate domain names into IP addresses. "#53" indicates that we are communicating with it on port 53, which is the standard port number domain name servers use to accept queries. Below this, we have our lookup information for microsoft.com. Our name server returned two entries, 134.170.185.46 and 134.170.188.221. This indicates that microsoft.com uses a round robin setup to distribute server load. When you accessmicrsoft.com, you may be directed to either of these servers and your packets will be routed to the correct destination. You can see that we have received a "Non-authoritative answer" to our query. An answer is "authoritative" only if our DNS has the complete zone file information for the domain in question. More often, our DNS will have a cache of information representing the last authoritative answer it received when it made a similar query, this information is passed on to you, but the server qualifies it as "non-authoritative": the information was recently received from an authoritative source, but the DNS server is not itself that authority.

5. **nmap :** Nmap (Network Mapper) is a security scanner originally written by Gordon Lyon (also known by his pseudonym Fyodor Vaskovich) used to discover hosts and services on a computer network, thus creating a "map" of the network. To accomplish its goal, Nmap sends specially crafted packets to the target host and then analyzes the responses. Unlike many simple port scanners that just send packets at some predefined constant rate, Nmap accounts for the network conditions (latency fluctuations, network congestion, the target interference with the scan) during the run. Also, owing to the large and active user community providing feedback and contributing to its features, Nmap has been able to extend its discovery capabilities beyond simply figuring out whether a host is up or down and which ports are open and closed; it can

determine the operating system of the target, names and versions of the listening services, estimated uptime, type of device, and presence of a firewall.

Nmap features include:

- Host Discovery – Identifying hosts on a network. For example, listing the hosts which respond to pings or have a particular port open.
- Port Scanning – Enumerating the open ports on one or more target hosts.
- Version Detection – Interrogating listening network services listening on remote devices to determine the application name and version number.
- OS Detection – Remotely determining the operating system and some hardware characteristics of network devices.

Basic commands working in Nmap:

- For target specifications: nmap<target‘s URL or IP with spaces between them>
- For OS detection: nmap -O <target-host's URL or IP>
- For version detection: nmap -sV<target-host's URL or IP>

SYN scan is the default and most popular scan option for good reasons. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by restrictive firewalls. It is also relatively unobtrusive and stealthy since it never completes TCP connections

Algorithm\Implementation Steps\Installation Steps -

- Installing Nmap from the link.
  sudo apt-get install nmap
- Obtaining Your IP addresses.
  Use the ifconfig command in Linux.
- Performing a Scan of the Local Network.

☐ For the following steps, please use the nmap command line tool installed on Ubuntu.

☐ Scan your subnet to determine how many hosts can be found. For example, if you are on the 192.168.1.0 subnet, you would enter the following command: nmap –sP 192.168.1.*

1. What is your subnet? _____
2. How many hosts were found? ____

☐ Next perform a stealth scan (Please use the IP for your subnet): nmap –sS –P0 –p 192.169.1.*

☐ Now, you’ll perform an OS identification. Use the Linux O/S to scan your Windows machine:

1. nmap –O Windows_IP_ADDRESS
2. OS Type 1:
3. Now we want to use the Windows machine to scan the Linux O/S. Go to a Windows DOS prompt and enter the following command:
4. nmap –O Linux_IP_ADDRESS
5. Now we will perform a service selection scan. Let’s scan for all computers with FTP running. We would do that as follows: nmap –p21 192.168.1.*

☐ List the IP addresses with that has the FTP open: _____

Input and Output -

- Installation of nmap:
  sudo apt-get install nmap
- nmap -sP 10.0.0.0/24
  Ping scans the network, listing machines that respond to ping.
- FIN scan (-sF)
  Sets just the TCP FIN bit.
- -sV (Version detection).
  Enables version detection, as discussed above. Alternatively, can use -A, which enables version detection among other things.
- -sO (IP protocol scan).
  IP protocol scan allows you to determine which IP protocols (TCP, ICMP, IGMP, etc.) are supported by target machines. This isn't technically a port scan, since it cycles through IP protocol numbers rather than TCP or UDP port numbers.
- -O (Enable OS detection) .
  Enables OS detection, as discussed above. Alternatively, you can use -A to enable OS detection along with other things.
- -p port ranges (Only scan specific ports).
  This option specifies which ports you want to scan and overrides the default Individual port numbers are OK, as are ranges separated by a hyphen (e.g. 1-1023). The beginning and/or end values of a range may be omitted, causing Nmap to use 1 and 65535, respectively.
- --top-ports <integer of 1 or greater>
  Scans the N highest-ratio ports found in nmap-services file.
- nmap –iflist
  host interface and route information with nmap by using ——iflistll option.

Conclusion -

Various reconnaissance tools are studied and used to gather primary network information.

Experiment No. 07

Aim - To Study packet sniffer tools wireshark, :-

1. Observer performance in promiscuous as well as non-promiscuous mode.
2. Show the packets can be traced based on different filters

Theory -

Wireshark, a network analysis tool formerly known as Ethereal, captures packets in real time and displays them in human-readable format. Wireshark includes filters, color-coding and other features that let you dig deep into network traffic and inspect individual packets.

Features of Wireshark:

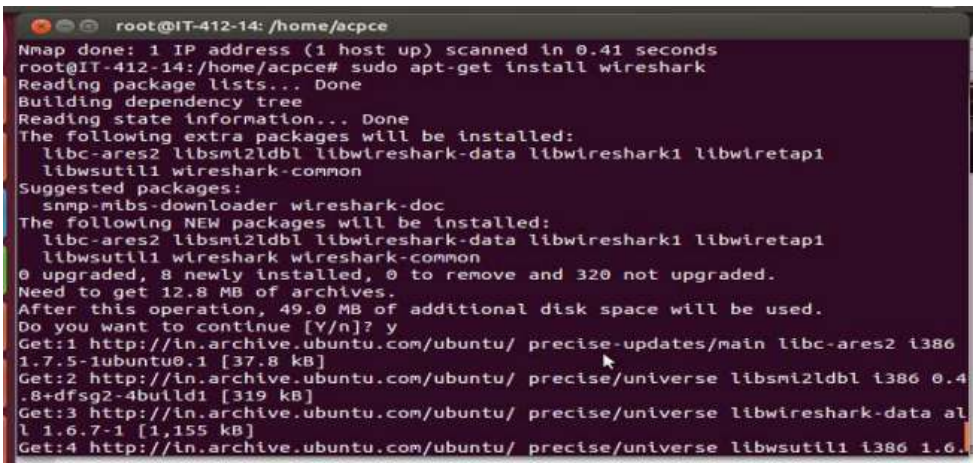
- Available for UNIX and Windows.
- Capture live packet data from a network interface.
- Open files containing packet data captured with tcpdump/WinDump, Wireshark, and a
- number of other packet capture programs.
- Import packets from text files containing hex dumps of packet data.
- Display packets with very detailed protocol information.
- Export some or all packets in a number of capture file formats.
- Filter packets on many criteria.
- Search for packets on many criteria.
- Colorize packet display based on filters.
- Create various statistics.

❖ Capturing Packets

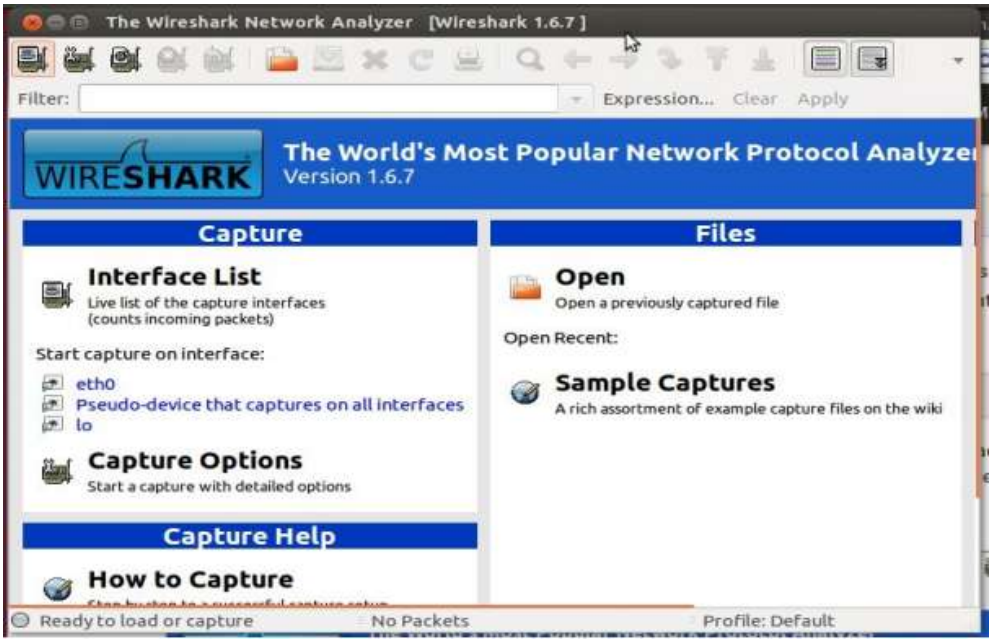
After downloading and installing wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface. You can configure advanced features by clicking Capture Options.

Installation of Wireshark -

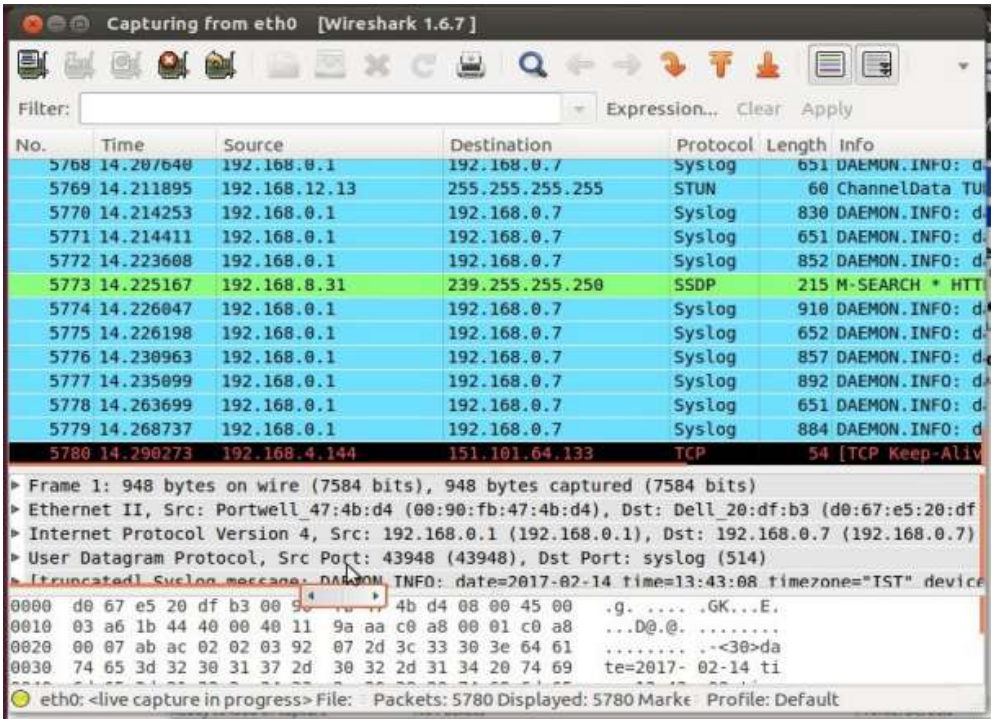
sudo apt-get install wireshark



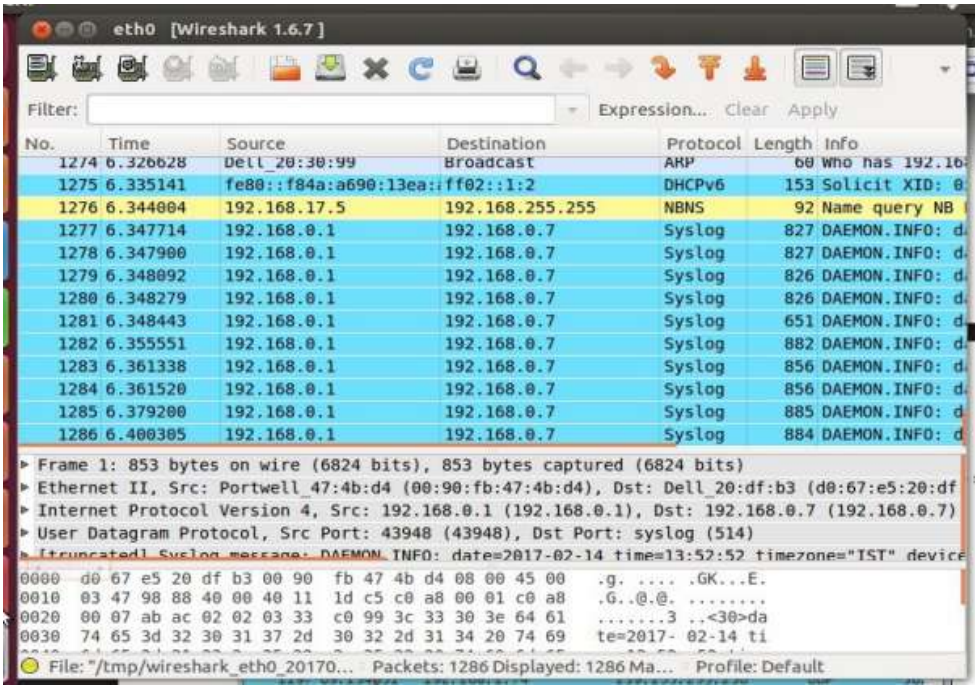
After downloading and installing wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface. You can configure advanced features by clicking Capture Options.



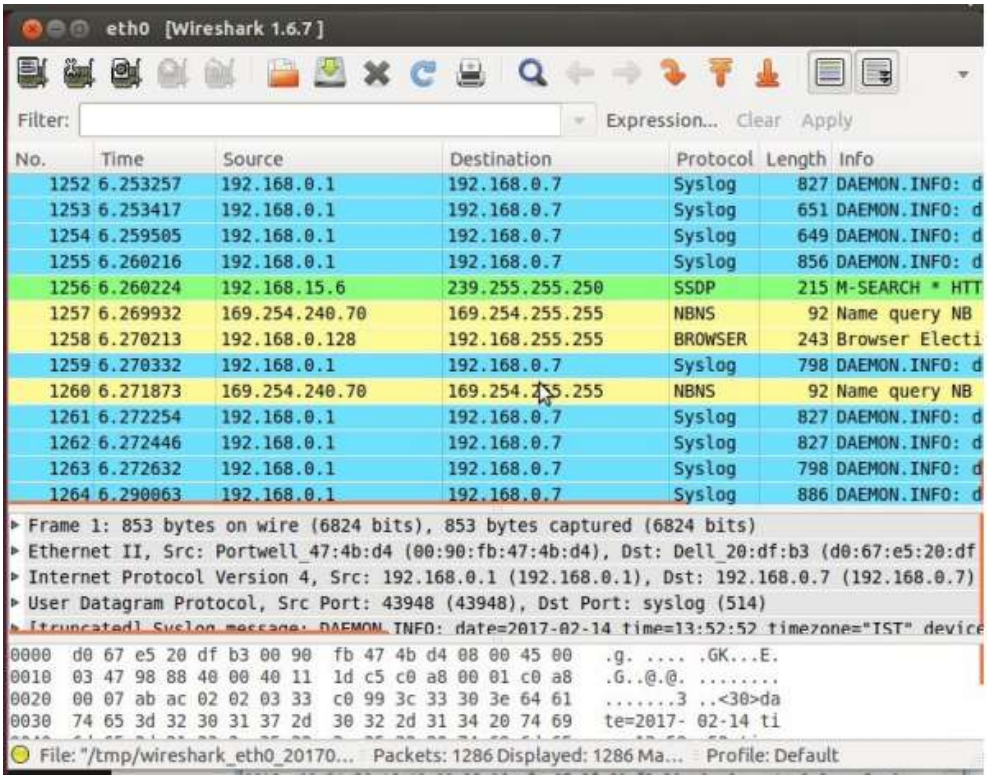
As soon as you click the interface‘s name, you‘ll see the packets start to appear in real time. Wireshark captures each packet sent to or from your system. If you‘re capturing on a wireless interface and have promiscuous mode enabled in your capture options, you‘ll also see other the other packets on the network



Click the stop capture button near the top left corner of the window when you want to stop capturing traffic.



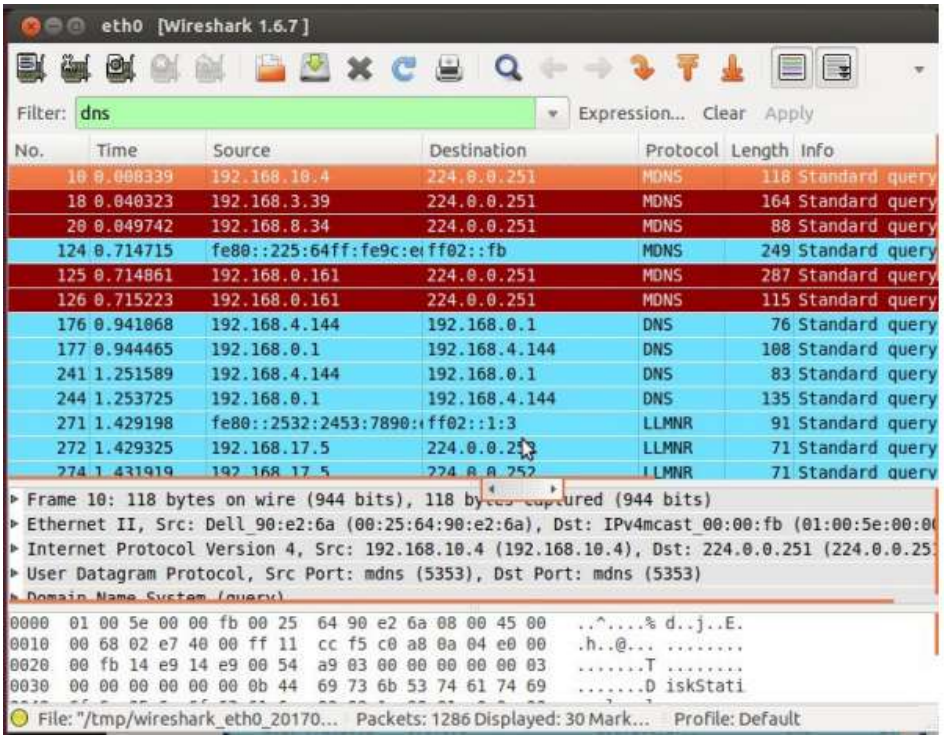
Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and black identifies TCP packets with problems — for example, they could have been delivered out-of-order.



❖ **Filtering Packets**

If you‘re trying to inspect something specific, such as the traffic a program sends when phoning home, it helps to close down all other applications using the network so you can narrow down the traffic. Still, you‘ll likely have a large amount of packets to sift through. That‘s where Wireshark‘s filters come in.

The most basic way to apply a filter is by typing it into the filter box at the top of the window and clicking Apply (or pressing Enter). For example, type —dnsll and you‘ll see only DNS packets. When you start typing, Wireshark will help you autocomplete your filter.



Conclusion -

We Studied packet sniffer tools wireshark.

Detailed information about packets is explored by applying filters.

Experiment No. 08

Aim - To Download & install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, etc. .

Theory -

Nmap

Nmap (Network Mapper) is a security scanner originally written by Gordon Lyon (also known by his pseudonym Fyodor Vaskovich) used to discover hosts and services on a computer network, thus creating a "map" of the network. To accomplish its goal, Nmap sends specially crafted packets to the target host and then analyzes the responses. Unlike many simple port scanners that just send packets at some predefined constant rate, Nmap accounts for the network conditions (latency fluctuations, network congestion, the target interference with the scan) during the run. Also, owing to the large and active user community providing feedback and contributing to its features, Nmap has been able to extend its discovery capabilities beyond simply figuring out whether a host is up or down and which ports are open and closed; it can determine the operating system of the target, names and versions of the listening services, estimated uptime, type of device, and presence of a firewall.

Nmap features include:

- Host Discovery – Identifying hosts on a network. For example, listing the hosts which respond to pings or have a particular port open.
- Port Scanning – Enumerating the open ports on one or more target hosts.
- Version Detection – Interrogating listening network services listening on remote devices to determine the application name and version number.
- OS Detection – Remotely determining the operating system and some hardware characteristics of network devices.

Basic commands working in Nmap:

- For target specifications: nmap<target‘s URL or IP with spaces between them>
- For OS detection: nmap -O <target-host's URL or IP>
- For version detection: nmap -sV<target-host's URL or IP>

SYN scan is the default and most popular scan option for good reasons. It can be performed quickly, scanning thousands of ports per second on a fast network not hampered by restrictive firewalls. It is also relatively unobtrusive and stealthy since it never completes TCP connections

Algorithm\Implementation Steps\Installation Steps -

- Installing Nmap from the link.
  sudo apt-get install nmap
- Obtaining Your IP addresses.
  Use the ifconfig command in Linux.
- Performing a Scan of the Local Network.

- ☐ For the following steps, please use the nmap command line tool installed on Ubuntu.
- ☐ Scan your subnet to determine how many hosts can be found. For example, if you are on the 192.168.1.0 subnet, you would enter the following command: nmap –sP 192.168.1.*
  3. What is your subnet? _____
  4. How many hosts were found? ____
- ☐ Next perform a stealth scan (Please use the IP for your subnet): nmap –sS –P0 –p 192.169.1.*
- ☐ Now, you'll perform an OS identification. Use the Linux O/S to scan your Windows machine:
  6. nmap –O Windows_IP_ADDRESS
  7. OS Type 1:
  8. Now we want to use the Windows machine to scan the Linux O/S. Go to a Windows DOS prompt and enter the following command:
  9. nmap –O Linux_IP_ADDRESS
  10. Now we will perform a service selection scan. Let's scan for all computers with FTP running. We would do that as follows: nmap –p21 192.168.1.*
- ☐ List the IP addresses with that has the FTP open: _____

Input and Output -

- Installation of nmap:
  sudo apt-get install nmap
- nmap -sP 10.0.0.0/24
  Ping scans the network, listing machines that respond to ping.
- FIN scan (-sF)
  Sets just the TCP FIN bit.
- -sV (Version detection).
  Enables version detection, as discussed above. Alternatively, can use -A, which enables version detection among other things.
- -sO (IP protocol scan).
  IP protocol scan allows you to determine which IP protocols (TCP, ICMP, IGMP, etc.) are supported by target machines. This isn't technically a port scan, since it cycles through IP protocol numbers rather than TCP or UDP port numbers.
- -O (Enable OS detection) .
  Enables OS detection, as discussed above. Alternatively, you can use -A to enable OS detection along with other things.
- -p port ranges (Only scan specific ports).
  This option specifies which ports you want to scan and overrides the default Individual port numbers are OK, as are ranges separated by a hyphen (e.g. 1-1023). The beginning and/or end values of a range may be omitted, causing Nmap to use 1 and 65535, respectively.
- --top-ports <integer of 1 or greater>
  Scans the N highest-ratio ports found in nmap-services file.
- nmap –iflist
  host interface and route information with nmap by using ––iflistǁ option.

Conclusion -

Network mapper tool is studied and used to gather comprehensive system and network primary network information

Experiment No. 09

Aim - To Detect ARP spoofing using nmap and/or open source tool ARPWATCH & wireshark

Theory -

ARP spoofing is a technique used to intercept network traffic by forging ARP messages. To detect ARP spoofing, you can use tools like Nmap, ARPWATCH, and Wireshark.

- **Using Nmap:**
  1. Open a terminal and type "sudo nmap -sP <ip address>" to scan the network.
  2. Look for duplicate MAC addresses in the output.
  3. If you find any duplicate MAC addresses, it could be an indication of ARP spoofing.

- **Using ARPWATCH:**
  1. Install ARPWATCH on your system.
  2. Configure ARPWATCH to monitor the network.
  3. If ARPWATCH detects any ARP spoofing, it will log the event.

- **Using Wireshark:**
  1. Start Wireshark and capture packets on the network interface.
  2. Filter for ARP traffic by typing "arp" in the filter field.
  3. Look for ARP requests and replies that do not match the expected MAC address for the IP address.
  4. If you find any such packets, it could be an indication of ARP spoofing.

Nmap can be used to scan the entire network or a specific range of IP addresses to identify all the connected devices. Once you have a list of connected devices, you can use Nmap to scan for open ports and services on each device to identify any potential vulnerabilities.

ARPWATCH can be configured to send alerts when it detects ARP spoofing, which can help you respond quickly to potential security threats. Additionally, ARPWATCH can be used to track MAC address changes over time, which can help you identify patterns of behavior that may indicate malicious activity.

Wireshark is a powerful tool for capturing and analyzing network traffic. In addition to filtering for ARP traffic, Wireshark can also be used to filter for other types of traffic, such as HTTP, SMTP, and FTP. This can help you identify potential security threats and monitor network activity in real-time.

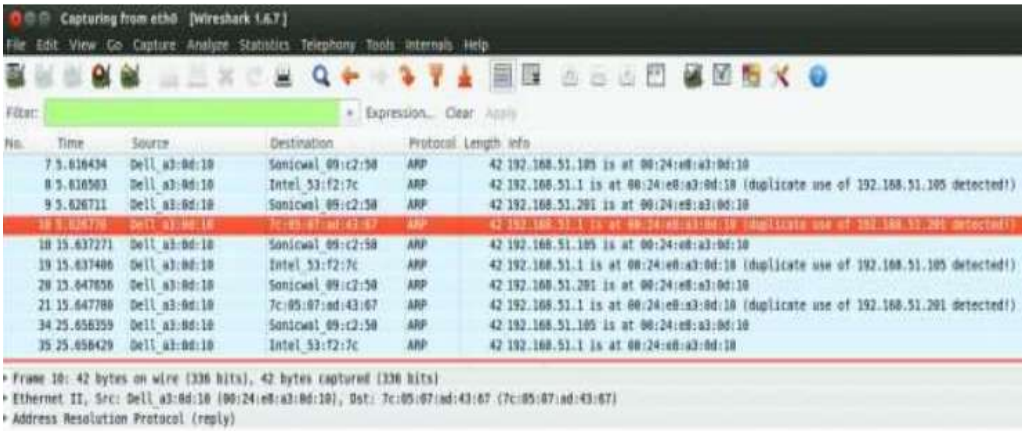
In addition to using these tools to detect ARP spoofing, it's also important to implement other security measures, such as using firewalls, implementing access controls, and regularly updating your software and security patches. By taking a comprehensive approach to network security, you can reduce the risk of ARP spoofing and other security threats.

Output -

Step 1 - Successful ARP Poisoning

```
Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.51.105
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.51.1

C:\Documents and Settings\rajesh>arp -a

Interface: 192.168.51.105 --- 0x10004
  Internet Address      Physical Address      Type
  192.168.51.1          00-17-c5-09-c2-50     dynamic
  192.168.51.110        00-1e-a6-25-a8-77     dynamic
  192.168.51.201        7c-05-07-ad-43-67     dynamic

C:\Documents and Settings\rajesh>arp -a

Interface: 192.168.51.105 --- 0x10004
  Internet Address      Physical Address      Type
  192.168.51.1          00-24-e8-a3-0d-10     dynamic
  192.168.51.204        00-24-e8-a3-0d-10     dynamic

C:\Documents and Settings\rajesh>_
```

Step 2 - Wireshark Capture on Attacker's PC-ARP Packets



Step 3 - Wireshark Capture on Attacker PC-Sniffed packets from Victim PC and Router

Conclusion -

We have successfully detected ARP spoofing using nmap and/or open source tool ARPWATCH and wireshark.

Experiment No. 10

Aim - To Use the NESSUS/ISO Kali Linux tool to scan the network for vulnerabilities

Theory -

Vulnerability scanning is an essential aspect of modern-day cybersecurity and Nessus is a well-known tool that provides a comprehensive solution for vulnerability assessments. It is a popular choice among security professionals and enthusiasts, due to its compatibility with Windows, MacOS, and Linux.

So how can you download and install Nessus on Kali, a widely-used penetration testing platform? With this step-by-step guide, you'll be up and running with Nessus in no time, equipped to proactively identify and mitigate vulnerabilities in your network.

- **What Is Nessus?**

Nessus, developed by Tenable Inc, is a widely-used open-source vulnerability scanner. It offers a paid subscription, Nessus Professional, as well as a free version, Nessus Essentials, which is limited to 16 IP addresses per scanner.

Nessus provides a range of services, including vulnerability assessments, network scans, web scans, asset discovery, and more, to aid security professionals, penetration testers, and other cybersecurity enthusiasts in proactively identifying and mitigating vulnerabilities in their networks.

How to Install Nessus on Kali Linux -

Unlike many security tools, Nessus doesn't come installed on Kali Linux. But it is very easy to download and install. Follow these steps to install Nessus on your Kali:

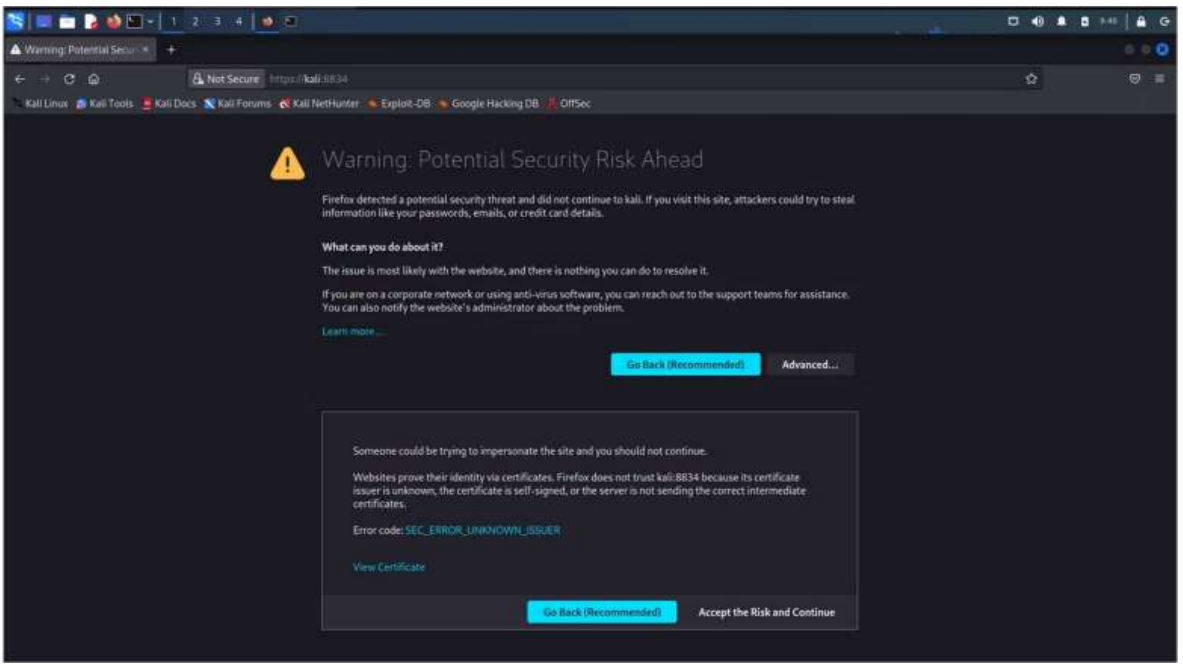
Step 1. Download the Nessus package for Debian on the Nessus website and make sure you set the Platform to **Linux-Debian-amd64**.

Step 2. When it's finished downloading, open your Linux terminal and navigate to the location you downloaded the Nessus file too.

Step 3. Install Nessus using this command: **dpkg -i Nessus-10.4.1-debian9_amd64.deb**

Step 4. Start the Nessus service with this command: systemctl **start** nessus

Step 5. On your browser, go to **https://kali:8834/.** It would show a warning page.



Step 6. Click on Advanced. Then, click on Accept Risk and Continue.

Step 7. Choose the Nessus Product you prefer. If you want the free version of Nessus, click on Nessus Essentials.

Step 8. Enter your name and email address to receive an activation code by email. Paste the activation code into the space provided and choose a username and password.

Step 9. Allow Nessus to download the necessary plugins.

Step 10. Once the plugin downloads have completed, you can start using the Nessus service.

Conclusion -

We have successfully used the NESSUS/ISO Kali Linux tool to scan the network for vulnerabilities

Experiment No. 11

Aim - To a) Set up IPSEC under LINUX.
b) Set up Snort and study the logs.
c) Explore the GPG tool of linux to implement email security

Theory -

The **GNU Privacy Guard** (GPG or gpg) tool is a native/baseos security tool for encrypting files. According to the gpg man page: gpg is the OpenPGP (Pretty Good Privacy) part of the GNU Privacy Guard (GnuPG). It is a tool to provide digital encryption and signing services using the OpenPGP standard. gpg features complete key management and all the bells and whistles you would expect from a full OpenPGP implementation.

The gpg utility has a lot of options, but fortunately for us, encrypting and decrypting are easy to do and only require that you know three options for quick use: Create or encrypt (-c), decrypt (-d), and extract and decrypt (no option).

- **Encrypting a file**

The quick method for encrypting a file is to issue the gpg command with the -c (create) option:

Encrypting a file with gpg leaves the original file intact, file1.txt, and adds the telltale .gpg extension to the newly encrypted file. You should probably remove the original file, file1.txt, so that the encrypted one is the sole source of the information contained in it. Alternatively, if you're going to share the encrypted version, you can rename it before sharing.
The .gpg extension isn't required, but it does let the user know which decryption tool to use to read the file. You can rename the file to anything you want.

- **Decrypting a file**

Decrypting a file means that you remove the encryption to read the file's contents. There's no extraction of content or creation of the original file when you decrypt.

Decrypting and extracting a file
If you want to extract the original file while decrypting it, strangely enough, you issue the gpg command with no options.

gpg has many more options than I've shown here. But these three are easy-to-use encryption and decryption options that will get you started protecting your files right away.

Output -

- **Encrypting a file:**

```
$ echo This is an encryption test > file1.txt

$ gpg -c file1.txt

 lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
 x Enter passphrase                                                   x
 x                                                                    x
 x                                                                    x
 x Passphrase: ***********_____________________________________       x
 x                                                                    x
 x       <OK>                                          <Cancel>       x
 mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj

 lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk
 x Please re-enter this passphrase                                    x
 x                                                                    x
 x Passphrase: ***********_____________________________________       x
 x                                                                    x
 x       <OK>                                          <Cancel>       x
 mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj

$ ls

file1.txt file1.txt.gpg
```

- **Decrypting a file**:

```
$ cat cfile.txt

This is an encryption and decryption test

$ gpg -c cfile.txt

< Set passphrase and repeat passphrase >

$ ls

$ cfile.txt cfile.txt.gpg

$ rm cfile.txt

$ gpg -d cfile.txt.gpg
gpg: AES encrypted data
gpg: encrypted with 1 passphrase
This is an encryption and decryption test

$ ls
cfile.txt.gpg

$ cat cfile.txt.gpg
c@yAw?D??^a??!s?????;??!?v9-3, ???XA??!?9v?}???
Ž??m??1./fK¹??R???:j?F?|?AS?O
```

- **Decrypting and extracting a file**:

```
$ ls
cfile.txt.gpg
$ gpg cfile.txt.gpg
< Passphrase prompt >
gpg: WARNING: no command supplied.  Trying to guess what you mean
...
gpg: AES encrypted data
gpg: encrypted with 1 passphrase
$ ls
cfile.txt  cfile.txt.gpg
```

Conclusion -

We have successfully a) set up IPSEC under LINUX. b) Setup up Snort and study the logs. & c) Explore the GPG tool of linux to implement email security.