DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATORY MANUAL

314456: Computer Network & Security Lab.

Sr. No.	Description
1.	Using a Network Simulator (e.g. packet tracer) configure a router using router commands, Access control lists, Routing Information Protocol & Network Address Translation.
2.	Using a Network Simulator (e.g. packet tracer) Configure EIGRP – Explore Neighbor-ship Requirements and Conditions, its K Values Metrics Assignment and Calculation, OSPF – Explore Neighbor-ship Condition and Requirement, Neighbor-ship states, OSPF Metric Cost Calculation, WLAN with static IP addressing and DHCP with MAC security and filters
3.	Socket Programming in C/C++ on Linux. TCP Client , TCP Server UDP Client , UDP Server
4.	Introduction to server administration (server administration commands and their applications) and configuration any three of below Server : (Study/Demonstration Only) FTP, Web Server.
5.	Implement a client and a server on different computers using python. Perform the communication between these two entities by using RSA cryptosystem.
6.	Implement a client and a server on different computers using python. Perform the authentication of sender between these two entities by using RSA digital signature cryptosystem.
7.	Implement a client and a server on different computers using python. Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.
8.	Use the snort intrusion detection package to analyze traffic and create a signature to identify problem traffic.

Network & Security Lab.			
Experiment No: 1	**Configure a router, Access control lists, Routing Information Protocol & Network Address Translation.**	**Page**	**1/6**

TITLE: Using a Network Simulator (e.g. packet tracer) configure a router using router commands, Access control lists, Routing Information Protocol & Network Address Translation.

OBJECTIVE: To learn fundamental concepts of configuration of router using router commands, Access control lists, Routing Information Protocol & Network Address Translation.

THEORY:

Router:

Definition: Routers are small physical devices that join multiple networks together. Technically, a router is a Layer 3 gateway device, meaning that it connects two or more networks and that the router operates at the network layer of the OSI model.

Home networks typically use a wireless or wired Internet Protocol (IP) router, IP being the most common OSI network layer protocol. An IP router such as a DSL or cable modem broadband router joins the home's local area network (LAN) to the wide-area network (WAN) of the Internet. A network router is a small electronic device that allows you build a home network simply. The home router serves as the core or "centerpiece" of the network to which computers, printers and other devices can be connected. Networking with a router helps you to (for example):

- share files between computers
- share an Internet connection between computers
- share a printer
- connect your game console or other home entertainment equipment to the Internet

Routers are not necessarily required to build a network. For example, you can connect two computers directly to each other with just a cable (or without wires in some cases). Home routers offer convenience and easier maintenance as your network grows.

Computer Network & Security Lab.			
Experiment No: 1	**Configure a router, Access control lists, Routing Information Protocol & Network Address Translation.**	**Page**	**2/6**

All of these networks rely on NAPs, backbones and **routers** to talk to each other. What is incredible about this process is that a message can leave one computer and travel halfway across the world through several different networks and arrive at another computer in a fraction of a second.

Commands for Configuring router:

Step 1

Example:
Router> **enable**
Router# **configure terminal**
Router(config)#

Step 2

hostname *name*
Example:
Router(config)# **hostname Router**
Router(config)#

Step 3

enable secret *password*
Example:
Router(config)# **enable secret cr1ny5ho**
Router(config)#

Step 4

enable password
Example:
Router(config)# **enable password Cisco**
Router(config)#

Step 5

Line vty
Example:
Router(config)#line vty 0 4
Router(config-line)# password Avcoe
Router(config-line)#exit
Router(config)#

Step 6

Line console
Example:
Router(config)#line console 0
Router(config-line)# password Avcoe
Router(config-line)#exit
Router(config)#

Step 7

ip address *ip-address mask*
Example:
Router(config-int)# **ip address 192.168.12.2 255.255.255.0**
Router(config-int)#

Step 8

no shutdown
Example:
Router(config-int)# **no shutdown**
Router(config-int)#

Step 9

exit
Example:
Router(config-int)# **exit**
Router(config)#

IP Access List:

This document describes how IP access control lists (ACLs) can filter network traffic. It also contains brief descriptions of the IP ACL types, feature availability, and an example of use in a network. An ACL is an ordered set of rules for filtering traffic. When the device determines that an ACL applies to a packet, it tests the packet against the rules. The first matching rule determines whether the packet is permitted or denied. If there is no match, the device applies a default rule. The device processes packets that are permitted and drops packets that are denied. You can use ACLs to protect networks and specific hosts from unnecessary or unwanted traffic. For example, you could use ACLs to disallow HTTP traffic from a high-security

network to the Internet. You could also use ACLs to allow HTTP traffic but only to specific sites, using the IP address of the site to identify it in an IP ACL.

ACL Types and Applications

IP ACLs—The device applies IPv4 ACLs only to IP traffic.

MAC ACLs—The device applies MAC ACLs only to non-IP traffic.

Configuring IP ACLs

Summary Steps:

1. **config t**
2. **[no] ip access-list** {*name* | **match-local-traffic**}
3. [*sequence-number*] {**permit | deny**} *protocol source destination*
4. **statistics per-entry**
5. **show ip access-lists** *name*
6. **copy running-config startup-config**

Configuring Routing Information Protocol:

RIP is a relatively old, but still commonly used, interior gateway protocol created for use in small, homogeneous networks. It is a classical distance-vector routing protocol. RIP is documented in RFC 1058.

RIP uses broadcast User Datagram Protocol (UDP) data packets to exchange routing information. The Cisco IOS software sends routing information updates every 30 seconds, which is termed advertising. If a router does not receive an update from another router for 180 seconds or more, it marks the routes served by the non-updating router as being unusable. If there is still no update after 240 seconds, the router removes all routing table entries for the non-updating router.

The metric that RIP uses to rate the value of different routes is hop count. The hop count is the number of routers that can be traversed in a route. A directly

connected network has a metric of zero; an unreachable network has a metric of 16. This small range of metrics makes RIP an unsuitable routing protocol for large networks. A router that is running RIP can receive a default network via an update from another router that is running RIP, or the router can source (generate) the default network itself with RIP. In both cases, the default network is advertised through RIP to other RIP neighbors.

Enabling RIP and Configuring RIP Parameters:

Perform the steps in this section to enable RIP and to configure RIP parameters.

Summary Steps:

1. enable
2. configure terminal
3. router rip
4. network ip-address
5. neighbor ip-address
6. offset-list [access-list-number | access-list-name] {in | out} offset[interface-type interface-number]
7. timers basic update invalid holddown flush [sleeptime]
8. end

NAT—Network Address Translation

IP addresses are scarce. An ISP might have a /16 (formerly class B) address, giving it 65,534 host numbers. If it has more customers than that, it has a problem. For home customers with dial-up connections, one way around the problem is to dynamically assign an IP address to a computer when it calls up and logs in and take the IP address back when the session ends. In this way, a single /16 address can handle up to 65,534 active users, which is probably good enough for an ISP with several hundred thousand customers. When the session is terminated, the IP address is reassigned to another caller. While this strategy works well for an ISP with a moderate number of home users, it fails for ISPs that primarily serve business customers.

The problem is that business customers expect to be on-line continuously

during business hours. Both small businesses, such as three-person travel agencies, and large corporations have multiple computers connected by a LAN. Some computers are employee PCs; others may be Web servers. Generally, there is a router on the LAN that is connected to the ISP by a leased line to provide continuous connectivity. This arrangement means that each computer must have its own IP address all day long. In effect, the total number of computers owned by all its business customers combined cannot exceed the number of IP addresses the ISP has. For a /16 address, this limits the total number of computers to 65,534. For an ISP with tens of thousands of business customers, this limit will quickly be exceeded.

Commands for Configuring NAT:

After Configuring Router:-

Step 1

Example:

Router> **enable**
Router# **configure terminal**
Router(config)#

Step 2

Example:

Router(config)# **ip nat inside source static** ***private ip outgoing ip***
Router(config)#

Step 3

Example:

Router(config)# **interface** ***incoming port on router***
Router(config-if)# **ip nat inside**

Step 4

Example:

Router(config)# **interface** ***outgoing port on router***
Router(config-if)# **ip nat outside**

Step 5

Example:

Router(config-if)#exit
Router(config)# Show ip nat

Conclusion: Thus we have implemented and Configure Router, ACL, RIP & NAT on Router.

Computer Network & Security Lab.			
Experiment No: 2	**Configure EIGRP, OSPF & WLAN.**	**Page**	**1/5**

TITLE: Using a Network Simulator (e.g. packet tracer) Configure EIGRP – Explore Neighbor-ship Requirements and Conditions, its K Values Metrics Assignment and Calculation, OSPF – Explore Neighbor-ship Condition and Requirement, Neighbor-ship states, OSPF Metric Cost Calculation, WLAN with static IP addressing and DHCP with MAC security and filters

OBJECTIVE: To learn fundamental concepts of configuration of router with EIGRP, OSPF and WLAN.

THEORY:

EIGRP:

EIGRP is an enhanced version of IGRP developed by Cisco Systems, Inc. EIGRP uses the same distance vector algorithm and distance information as IGRP. However, the convergence properties and the operating efficiency of EIGRP have improved significantly over IGRP.

The convergence technology is based on research conducted at SRI International and employs an algorithm referred to as the Diffusing Update Algorithm (DUAL). This algorithm guarantees loop-free operation at every instant throughout a route computation and allows all devices involved in a topology change to synchronize at the same time. Routers that are not affected by topology changes are not involved in recomputations. The convergence time with DUAL rivals that of any other existing routing protocol.

EIGRP offers the following features:

• **Fast convergence**—The DUAL algorithm allows routing information to converge as quickly as any currently available routing protocol.

• Partial updates—EIGRP sends incremental updates when the state of a destination changes, instead of sending the entire contents of the routing table. This feature minimizes the bandwidth required for EIGRP packets.

Less CPU usage than IGRP—This occurs because full update packets do not have to be processed each time they are received.

• **Neighbor discovery mechanism**—This is a simple hello mechanism used to learn

about neighboring routers. It is protocol-independent.
• Variable-length subnet masks
• Arbitrary route summarization
• Scaling—EIGRP scales to large networks.

EIGRP has the following four basic components:
• Neighbor discovery/recovery
• Reliable transport protocol
• DUAL finite state machine
• Protocol-dependent modules

Enable EIGRP

To create an EIGRP routing process, use the following commands, beginning in global configuration
mode:

Step Command

1 **router eigrp** *autonomous-system*

Enable an EIGRP routing process in global configuration mode.

2 **network** *network-number*

Associate networks with an EIGRP routing process in router configuration mode. EIGRP sends updates to the interfaces in the specified networks. If you do not specify an interface's network, it will not be advertised in any EIGRP update.

Optimal Shortest Path First (OSPF):-

Open Shortest Path First (OSPF) is a routing protocol developed for Internet Protocol (IP) networks by the Interior Gateway Protocol (IGP) working group of the Internet Engineering Task Force (IETF). The working group was formed in 1988 to design an IGP based on the Shortest Path First (SPF) algorithm for use in the Internet. Similar to the Interior Gateway Routing Protocol (IGRP), OSPF was created because in the mid-1980s, the Routing Information Protocol (RIP) was increasingly incapable of serving large, heterogeneous internetworks. This chapter examines the OSPF routing environment, underlying routing algorithm, and general protocol components.

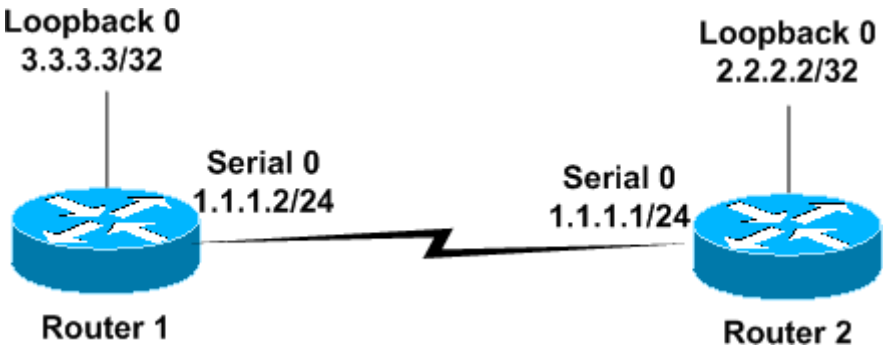
OSPF was derived from several research efforts, including Bolt, Beranek, and Newman's (BBN's) SPF algorithm developed in 1978 for the ARPANET (a landmark packet-switching network developed in the early 1970s by BBN), Dr. Radia Perlman's research on fault-tolerant broadcasting of routing information (1988), BBN's work on area routing (1986), and an early version of OSI's Intermediate System-to-Intermediate System (IS-IS) routing protocol.

OSPF has two primary characteristics. The first is that the protocol is open, which means that its specification is in the public domain. The OSPF specification is published as Request For Comments (RFC) 1247. The second principal characteristic is that OSPF is based on the SPF algorithm, which sometimes is referred to as the Dijkstra algorithm, named for the person credited with its creation.

OSPF is a link-state routing protocol that calls for the sending of link-state advertisements (LSAs) to all other routers within the same hierarchical area. Information on attached interfaces, metrics used, and other variables is included in OSPF LSAs. As OSPF routers accumulate link-state information, they use the SPF algorithm to calculate the shortest path to each node.

Computer Network & Security Lab.			
Experiment No: 2	**Configure EIGRP, OSPF & WLAN.**	**Page**	**4/5**

Network Diagram



Router1

```
Router1
!
interface Loopback0
  ip address 3.3.3.3 255.255.255.255
!
interface Serial0
 ip address 1.1.1.2 255.255.255.0
!
router ospf 1
network 1.1.1.0 0.0.0.255 area 0

!--- Configures the Serial Interface S0
under OSPF area 0.
!
```

Router2

```
Router2
!
interface Loopback0
 ip address 2.2.2.2 255.255.255.255
interface Serial0
 ip address 1.1.1.1 255.255.255.0
 clockrate 2000000
router ospf 1
network 1.1.1.0 0.0.0.255 area 0
```

Computer Network & Security Lab.			
Experiment No: 2	**Configure EIGRP, OSPF & WLAN.**	**Page**	**5/5**

Wireless LAN:

A wireless local-area network (WLAN) is a group of colocated computers or other devices that form a network based on radio transmissions rather than wired connections. A Wi-Fi network is a type of WLAN; anyone connected to Wi-Fi while reading this webpage is using a WLAN. A wireless LAN (WLAN) is a wireless computer network that links two or more devices using wireless communication to form a local area network (LAN) within a limited area such as a home, school, computer laboratory, campus, or office building. This gives users the ability to move around within the area and remain connected to the network. Through a gateway, a WLAN can also provide a connection to the wider Internet.

Wireless LANs based on the IEEE 802.11 standards are the most widely used computer networks in the world. These are commonly called Wi-Fi, which is a trademark belonging to the Wi-Fi Alliance. They are used for home and small office networks that link together laptop computers, printers, smartphones, Web TVs and gaming devices with a wireless router, which links them to the internet. Hotspots provided by routers at restaurants, coffee shops, hotels, libraries, and airports allow consumers to access the internet with portable wireless devices.

By allowing work to happen anywhere, wireless networks don't simply increase productivity and provide convenience. They can redefine enterprise goals and how they are achieved—not just in offices but also in factories, healthcare facilities, and schools.

How does a WLAN work?

Like broadcast media, a WLAN transmits information over radio waves. Data is sent in packets. The packets contain layers with labels and instructions that, along with the unique MAC (Media Access Control) addresses assigned to endpoints, enable routing to intended locations.

Conclusion: Thus we have implemented and Configure EIGRP, OSPF & WLAN.

Computer Network & Security Lab.			
Experiment No: 3	**Socket Programming in C/C++ on Linux.**	**Page**	**1/3**

TITLE: Socket Programming in C/C++ on Linux. TCP Client, TCP Server UDP Client , UDP Server.

OBJECTIVE: To learn concepts of Socket Programming in C/C++ on Linux. TCP Client, TCP Server, UDP Client, UDP Server.

THEORY: In UDP, the client does not form a connection with the server like in TCP and instead just sends a datagram. Similarly, the server need not accept a connection and just waits for datagrams to arrive. Datagrams upon arrival contain the address of the sender which the server uses to send data to the correct client.

The entire process can be broken down into the following steps :

UDP Server :

1. Create a UDP socket.
2. Bind the socket to the server address.
3. Wait until the datagram packet arrives from the client.
4. Process the datagram packet and send a reply to the client.
5. Go back to Step 3.

UDP Client :

1. Create a UDP socket.
2. Send a message to the server.
3. Wait until response from the server is received.
4. Process reply and go back to step 2, if necessary.
5. Close socket descriptor and exit.

Necessary Functions :

int socket(int domain, int type, int protocol)
Creates an unbound socket in the specified domain.

Computer Network & Security Lab.			
Experiment No: 3	**Socket Programming in C/C++ on Linux.**	**Page**	**2/3**

Server/Client Applications:

The basic mechanisms of client-server setup are:

1. A client app send a request to a server app.
2. The server app returns a reply.
3. Some of the basic data communications between client and server are:
    1. File transfer - sends name and gets a file.
    2. Web page - sends url and gets a page.
    3. Echo - sends a message and gets it back.

Server Socket:

1. create a socket - Get the file descriptor!
2. bind to an address -What port am I on?
3. listen on a port, and wait for a connection to be established.
4. accept the connection from a client.
5. send/recv - the same way we read and write for a file.
6. shutdown to end read/write.
7. close to releases data.

Client Socket:

1. create a socket.
2. bind* - this is probably be unnecessary because you're the client, not the server.
3. connect to a server.
4. send/recv - repeat until we have or receive data

Computer Network & Security Lab.			
Experiment No: 3	**Socket Programming in C/C++ on Linux.**	**Page**	**3/3**

5. shutdown to end read/write.
6. close to releases data.

Conclusion: Thus we have implemented Socket Programming in C/C++ on Linux.

Computer Network & Security Lab.			
Experiment No: 4	**Introduction to server administration.**	**Page**	**1/3**

TITLE: Introduction to server administration (server administration commands and their applications) and configuration any three of below Server: (Study/Demonstration Only) FTP, Web Server.

OBJECTIVE: To study server administration (server administration commands and their applications)

THEORY:

FTP Server:

The primary purpose of an FTP server is to allow users to upload and download files.An FTP server is a computer that has a file transfer protocol (FTP) address and is dedicated to receiving an FTP connection. FTP is a protocol used to transfer files via the internet between a server (sender) and a client (receiver)**.** An FTP server is a computer that offers files available for download via an FTP protocol, and it is a common solution used to facilitate remote data sharing between computers.
An FTP server is an important component in FTP architecture and helps in exchanging files over the internet. The files are generally uploaded to the server from a personal computer or other removable hard drives (such as a USB flash drive) and then sent from the server to a remote client via the FTP protocol.
An FTP server needs a TCP/IP network to function and is dependent on the use of dedicated servers with one or more FTP clients. In order to ensure that connections can be established at all times from the clients, an FTP server is usually switched on; up and running 24/7.
An FTP server is also known as an FTP site or FTP host.

Although the FTP server actually sends files over the internet, it generally acts as the midpoint between the real sender of a file and its recipient. The recipient must access the server address, which can either be a URL (e.g., ftp://exampleserver.net) or as a numeric address (usually the IP address of the server). All file transfer protocol site addresses begin with ftp://. FTP servers usually listen for client connections on port 21 since the FTP protocol generally uses this port as its principle route of communication. FTP runs on two different Transmission Control Protocol ports: 20 and 21. FTP ports 20 and 21 must both be open on the network for successful file transfers.

Computer Network & Security Lab.			
Experiment No: 4	**Introduction to server administration.**	**Page**	**2/3**

The FTP server allows the downloading and uploading of files. The FTP server's administrator can restrict access for downloading different files and from different folders residing in the FTP server. Files residing in FTP servers can be retrieved by common web browsers, but they may not support protocol extensions like FTPS. With an FTP connection, it is possible to resume an interrupted download that was not successfully completed; in other words, checkpoint restart support is provided. For the client to establish a connection to the FTP server, the username and password are sent using USER and PASS commands. Once accepted by the FTP server, an acknowledgment is sent to the client and the session can start. Failure to open both ports 20 & 21 prevents the full back-and-forth transfer from being made.
The FTP server can provide connection to users without login credentials; however, the FTP server can authorize these to have only limited access. FTP servers can also provide anonymous access. This access allows users to download files from the servers anonymously but prohibits uploading files to FTP servers.
Beyond routine file transfer operations, FTP servers are also used for offsite backup of critical data. FTP servers are quite inexpensive solutions for both data transfer and backup operations, especially if security is not a concern. However, when simple login and authentication features are not sufficient to guarantee an adequate degree of security (such as when transferring sensitive or confidential information), two secure file transfer protocol alternatives, SFTP and FTP/S, are also available. These secure FTP server options offer additional levels of security such as data encryption.

Web Server:

The term *web server* can refer to hardware or software, or both of them working together.

1. On the hardware side, a web server is a computer that stores web server software and a website's component files. (for example, HTML documents, images, CSS stylesheets, and JavaScript files) A web server connects to the Internet and supports physical data interchange with other devices connected to the web.
2. On the software side, a web server includes several parts that control how web users access hosted files. At a minimum, this is an *HTTP server*. An HTTP

Computer Network & Security Lab.			
Experiment No: 4	**Introduction to server administration.**	**Page**	**3/3**

3. server is software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). An HTTP server can be accessed through the domain names of the websites it stores, and it delivers the content of these hosted websites to the end user's device.

At the most basic level, whenever a browser needs a file that is hosted on a web server, the browser requests the file via HTTP. When the request reaches the correct (hardware) web server, the (software) *HTTP server* accepts the request, finds the requested document, and sends it back to the browser, also through HTTP. (If the server doesn't find the requested document, it returns a 404 response instead.)

To publish a website, you need either a static or a dynamic web server.

A static web server, or stack, consists of a computer (hardware) with an HTTP server (software). We call it "static" because the server sends its hosted files as-is to your browser.

A dynamic web server consists of a static web server plus extra software, most commonly an application server and a database. We call it "dynamic" because the application server updates the hosted files before sending content to your browser via the HTTP server.

For example, to produce the final webpages you see in the browser, the application server might fill an HTML template with content from a database. Sites like MDN or Wikipedia have thousands of webpages. Typically, these kinds of sites are composed of only a few HTML templates and a giant database, rather than thousands of static HTML documents. This setup makes it easier to maintain and deliver the content.

Conclusion: Thus we studied server administration (server administration commands and their applications).

Computer Network & Security Lab.			
Experiment No : 5	**Implement Client Server and communication using RSA cryptosystem.**	**Page**	**1/5**

TITLE :

Implement a client and a server on different computers using python. Perform the communication between these two entities by using RSA cryptosystem.

OBJECTIVE :

To establish client server communication and RSA algorithm.

THEORY:

Bind():

1. The **bind()** method of Python's socket class assigns an **IP address** and **a port number** to a socket instance.
2. The **bind()** method is used when a socket needs to be made a server socket.
3. As server programs listen on published ports, it is required that a **port** and the **IP address** to be assigned explicitly to a server socket.

Listen():

1. Calling listen() makes a socket ready for accepting connections.
2. The listen() method should be called before calling the accept() method on the server socket.
3. The listen() function accepts a queue size through the parameter backlog. This denotes maximum number of connections that can be queued for this socket by the operating system. Once 'backlog' number of connections is in the socket's queue, the kernel will reject incoming connections to the socket.

Accept ():

1. The accept() method of Python's socket class, accepts an incoming connection request from a TCP client.
2. The accept() method is called on a TCP based server socket.

3. When connect() is called at the client side with the IP address and port number of the server, the connect request is received with the accept() call at the server side.

4. Upon accepting a connection request from a TCP based client, the accept() method called on the server socket returns a socket that is connected to the client.

5. Data can be sent and received using the socket returned by the accept() method.

Server :

A server has a bind() method which binds it to a specific IP and port so that it can listen to incoming requests on that IP and port. A server has a listen() method which puts the server into listening mode. This allows the server to listen to incoming connections. And last a server has an accept() and close() method. The accept method initiates a connection with the client and the close method closes the connection with the client.

- First of all, import socket which is necessary.
- Then made a socket object and reserved a port on our pc.
- After that, bound server to the specified port. Passing an empty string means that the server can listen to incoming connections from other computers as well. If it would have passed 127.0.0.1 then it would have listened to only those calls made within the local computer.
- After that put the server into listening mode 5 here means that 5 connections are kept waiting if the server is busy and if a 6th socket tries to connect then the connection is refused.
- At last, make a while loop and start to accept all incoming connections and close those connections after a thank you message to all connected sockets.

Client :

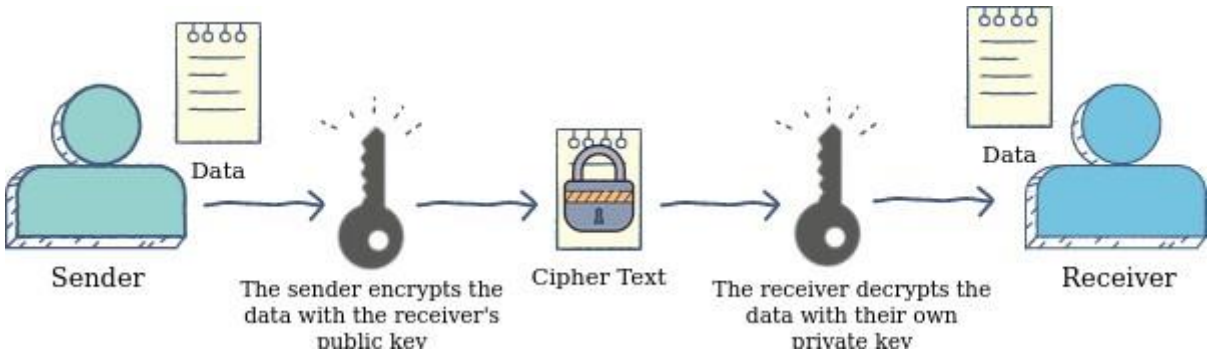
- First of all, make a socket object.
- Then connect to localhost on port 12345 (the port on which our server runs) and lastly, receive data from the server and close the connection.

Computer Network & Security Lab.			
Experiment No : 5	**Implement Client Server and communication using RSA cryptosystem.**	**Page**	**3/5**

RSA Cryptosystem:

The **RSA algorithm** is an asymmetric cryptography algorithm; this means that it uses a *public* key and a *private* key (i.e two different, mathematically linked keys). As their names suggest, a public key is shared publicly, while a private key is secret and must not be shared with anyone.

The following illustration highlights how asymmetric cryptography works:



Algorithm

The RSA algorithm holds the following features −

- RSA algorithm is a popular exponentiation in a finite field over integers including prime numbers.
- The integers used by this method are sufficiently large making it difficult to solve.
- There are two sets of keys in this algorithm: private key and public key.

Step 1: Generate the RSA modulus

The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown −

N=p*q

Here, let N be the specified large number.

Step 2: Derived Number (e)

Consider number e as a derived number which should be greater than 1 and less than (p-1) and (q-1). The primary condition will be that there should be no common factor of (p-1) and (q-1) except 1

Step 3: Public key

The specified pair of numbers **n** and **e** forms the RSA public key and it is made public.

Step 4: Private Key

Private Key **d** is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows −

ed = 1 mod (p-1) (q-1)

The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

Encryption Formula

Consider a sender who sends the plain text message to someone whose public key is **(n,e).** To encrypt the plain text message in the given scenario, use the following syntax −

C = Pe mod n

Decryption Formula

The decryption process is very straightforward and includes analytics for calculation in a systematic approach. Considering receiver **C** has the private key **d**, the result modulus will be calculated as −

Experiment No : 5	Implement Client Server and communication using RSA cryptosystem.	Page	5/5

Advantages of RSA

- **No Key Sharing:** RSA encryption depends on using the receiver's public key, so you don't have to share any secret key to receive messages from others.
- **Proof of Authenticity:** Since the key pairs are related to each other, a receiver can't intercept the message since they won't have the correct private key to decrypt the information.
- **Faster Encryption:** The encryption process is faster than that of the DSA algorithm.
- **Data Can't Be Modified:** Data will be tamper-proof in transit since meddling with the data will alter the usage of the keys. And the private key won't be able to decrypt the information, hence alerting the receiver of manipulation.

CONCLUSION:

Thus we have implemented the client server communication and RSA cryptosystem.

Computer Network & Security Lab.			
Experiment No : 6	**Implement Client Server and communication using RSA digital signature cryptosystem.**	**Page**	**1/5**

TITLE:

Implement a client and a server on different computers using python. Perform the authentication of sender between these two entities by using RSA digital signature cryptosystem.

OBJECTIVE:

To study authentication of sender between these two entities by using RSA digital signature cryptosystem.

THEORY:

Digital Signature:

As the name sounds are the new alternative to sign a document digitally. It ensures that the message is sent by the intended user without any tampering by any third party (attacker). In simple words, digital signatures are used to verify the authenticity of the message sent electronically.

RSA:

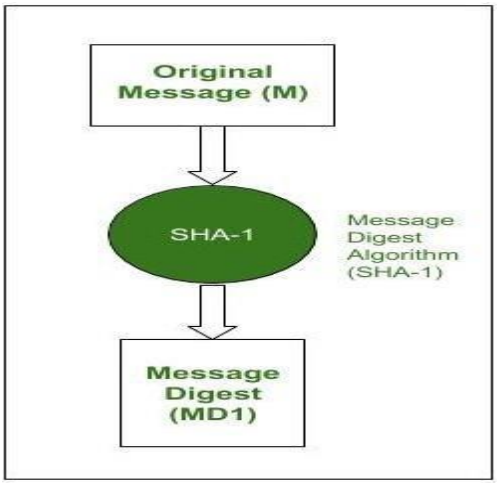
It is the most popular asymmetric cryptographic algorithm. It is primarily used for encrypting message s but can also be used for performing digital signature over a message.

Let us understand how RSA can be used for performing digital signatures step-by-step. Assume that there is a sender (A) and a receiver (B). A wants to send a message (M) to B along with the digital signature (DS) calculated over the message.

Step-1:

Sender A uses SHA-1 Message Digest Algorithm to calculate the message digest (MD1) over the original message M.

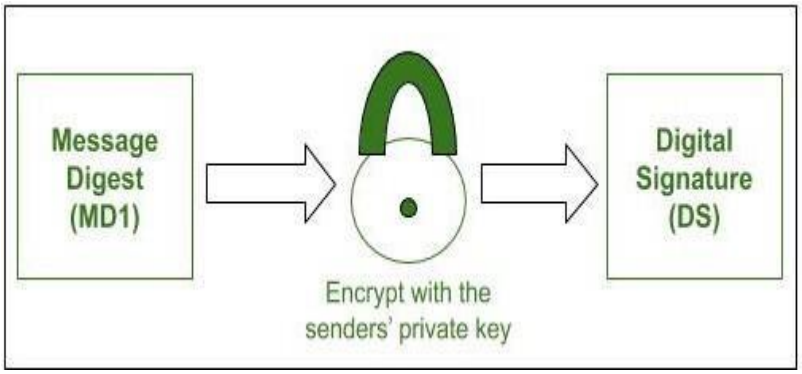
Computer Network & Security Lab.			
Experiment No : 6	**Implement Client Server and communication using RSA digital signature cryptosystem.**	**Page**	**2/5**



Message digest calculation

Step-2:

A now encrypts the message digest with its private key. The output of this process is called Digital Signature (DS) of A.

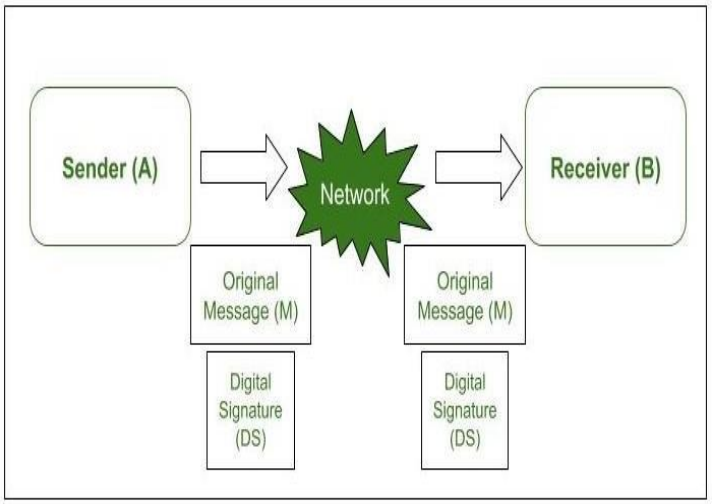


Digital signature creation

Step-3 :

Now sender A sends the digital signature (DS) along with the original message (M) to B.

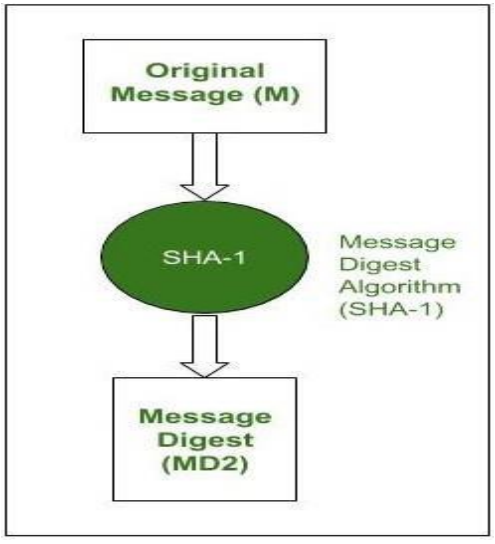
Computer Network & Security Lab.			
Experiment No : 6	**Implement Client Server and communication using RSA digital signature cryptosystem.**	**Page**	**3/5**



Transmission of original message and digital signature simultaneously

Step-4 :

When B receives the Original Message(M) and the Digital Signature(DS) from A, it first uses the same message-digest algorithm as was used by A and calculates its own Message Digest (MD2) for M.

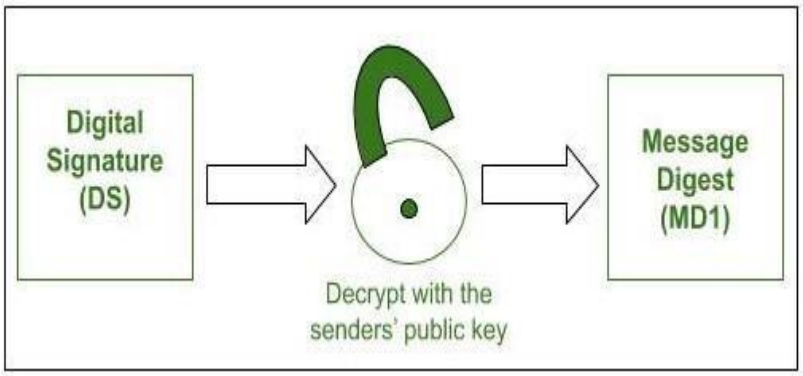


Receiver calculates its own message digest

Computer Network & Security Lab.			
Experiment No : 6	**Implement Client Server and communication using RSA digital signature cryptosystem.**	**Page**	**4/5**

Step-5 :

Now B uses A's public key to decrypt the digital signature because it was encrypted by A's private key. The result of this process is the original Message Digest (MD1) which was calculated by A.

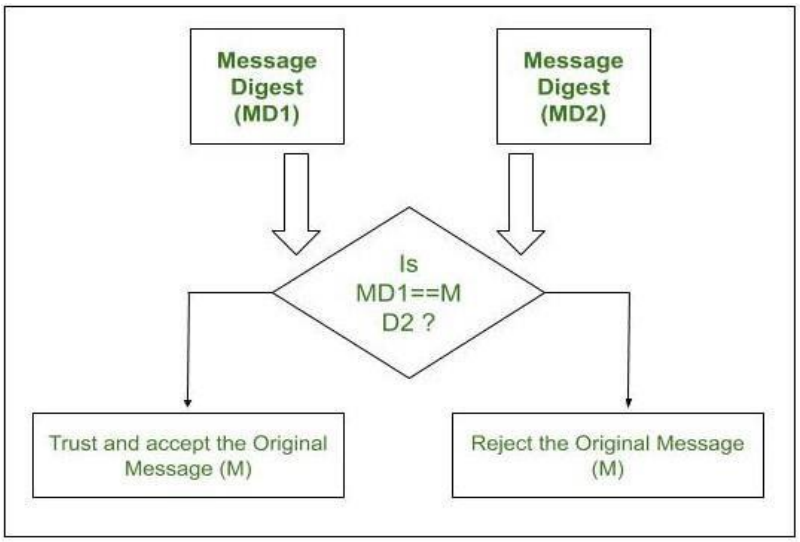


Receiver retrieves sender's message digest

Step-6 :

If MD1==MD2, the following facts are established as follows.

- B accepts the original message M as the correct, unaltered message from A.
- It also ensures that the message came from A and not someone posing as A.



Digital signature verification

The message digest (MD1) was encrypted using A's private key to produce a digital signature. Therefore, the digital signature can be decrypted using A's public key (due to asymmetric form of

RSA). If the receiver B is able to decrypt the digital signature using A's public key, it means that the message is received from A itself and now A cannot deny that he/she has not sent the message.

It also proves that the original message did not tamper because when the receiver B tried to find its own message digest MD2, it matched with that of A's MD1.

RSA Digital Signature Scheme: In RSA, d is private; e and n are public.

- Alice creates her digital signature using S=M^d mod n where M is the message
- Alice sends Message M and Signature S to Bob
- Bob computes M1=S^e mod n
- If M1=M then Bob accepts the data sent by Alice.

CONCLUSION :

Successfully implemented Client Server and communication using RSA digital signature cryptosystem.

Computer Network & Security Lab.			
Experiment No : 7	**Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.**	**Page**	**1/5**

TITLE:

Implement a client and a server on different computers using python. Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.

OBJECTIVE :

To learn the DES algorithm and Diffie Hellman key exchange algorithm.

THOERY:

DES :

The DES (Data Encryption Standard) algorithm is a symmetric-key block cipher created in the early 1970s by an IBM team and adopted by the National Institute of Standards and Technology (NIST). The algorithm takes the plain text in 64-bit blocks and converts them into ciphertext using 48-bit keys.Since it's a symmetric-key algorithm, it employs the same key in both encrypting and decrypting the data. If it were an asymmetrical algorithm, it would use different keys for encryption and decryption.

DES Algorithm Steps

To put it in simple terms, DES takes 64-bit plain text and turns it into a 64-bit ciphertext. And since we're talking about asymmetric algorithms, the same key is used when it's time to decrypt the text.

The algorithm process breaks down into the following steps:

1. The process begins with the 64-bit plain text block getting handed over to an initial permutation (IP) function.
2. The initial permutation (IP) is then performed on the plain text.
3. Next, the initial permutation (IP) creates two halves of the permuted block, referred to as Left Plain Text (LPT) and Right Plain Text (RPT).

Experiment No : 7	Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.	Page	2/5

4. Each LPT and RPT goes through 16 rounds of the encryption process.
5. Finally, the LPT and RPT are rejoined, and a Final Permutation (FP) is performed on the newly combined block.
   The result of this process produces the desired 64-bit ciphertext.

The encryption process step (step 4, above) is further broken down into five stages:

1. Key transformation
2. Expansion permutation
3. S-Box permutation
4. P-Box permutation
5. XOR and swap

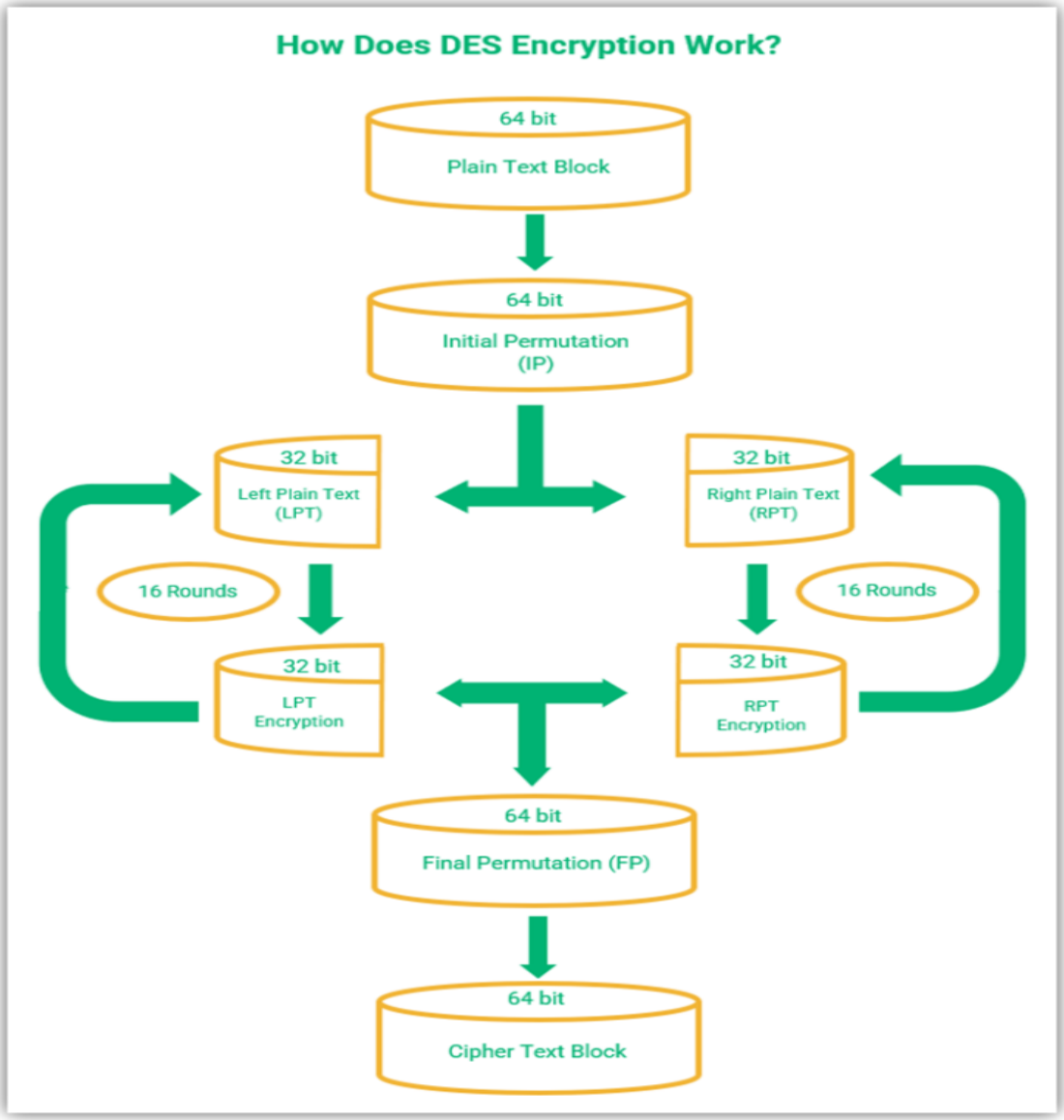
For decryption, we use the same algorithm, and we reverse the order of the 16 round keys.

DES Modes of Operation

Experts using DES have five different modes of operation to choose from.

- **Electronic Codebook (ECB).** Each 64-bit block is encrypted and decrypted independently
- **Cipher Block Chaining (CBC).** Each 64-bit block depends on the previous one and uses an Initialization Vector (IV)
- **Cipher Feedback (CFB).** The preceding ciphertext becomes the input for the encryption algorithm, producing pseudorandom output, which in turn is XORed with plaintext, building the next ciphertext unit
- **Output Feedback (OFB).** Much like CFB, except that the encryption algorithm input is the output from the preceding DES
- **Counter (CTR).** Each plaintext block is XORed with an encrypted counter. The counter is then incremented for each subsequent block

Computer Network & Security Lab.			
Experiment No : 7	**Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.**	**Page**	**3/5**



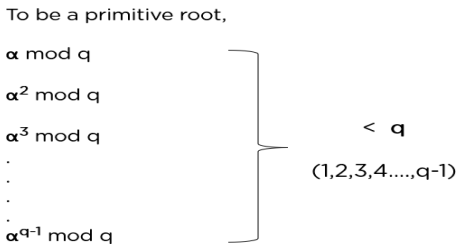
Experiment No : 7	Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.	Page	4/5

Diffie Hellman Key Exchange:

1. The Diffie-Hellman algorithm is a method for securely exchanging cryptographic keys over insecure channels without compromising the security and integrity of data transmission.
2. It was developed and published in 1976 by Martin Hellman and Whitefield Diffie. Until you received the asymmetric encryption algorithms that never relied on any category of key exchange, symmetric encryption was the only way to communicate securely.
3. A secure method to exchange the private keys for this brand of cryptography was much needed.

The steps needed for the Diffie-Hellman key exchange are as follows:

Step 1: You choose a prime number q and select a primitive root of q as α. To be a primitive root, it must satisfy the following criteria:



Step 2: Assume the private key for our sender as Xa where Xa < q. The public key can be calculated as Ya = αxa mod q. So, the key pair for your sender becomes {Xa, Ya}.

Assume the private key for the receiver to be Xb where Xb < q. The public key for the receiver is calculated as Yb = αxb mod q. For the receiver, the key pair becomes {Xb, Yb}.

Step 3: To generate the final secret key, you use three parameters. For the sender, you need the private key (Xa), the receiver's public key (Yb), and the original q. The formula to calculate the key is K = (Yb)Xa mod q.

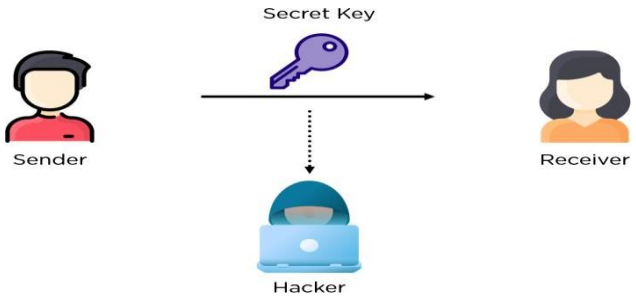
Experiment No : 7	Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.	Page	5/5

For the receiver, you need the private key (Ya), sender's public key (Xb), and the original q. The formula to calculate the secret key is K = (Ya)Xb mod q.

If both the values of K generated are equal, the Diffie-Hellman key exchange algorithm is complete.

Why Is the Diffie-Hellman Key Exchange Algorithm Necessary?

Symmetric encryption has always been a reliable method of cryptography for the exchange of private information. A glaring flaw has always been the difficulty in sharing the requisite secret key with the receiver of the message. It can intercept any key transmitted over an insecure channel by hackers, who can then use the same key to decrypt the encrypted ciphertexts.



The Diffie Hellman algorithm solves this problem using one-way functions that enable only the sender and receiver to decrypt the message using a secret key.

CONCLUSION:

Thus we have implemented the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.

Computer Network & Security Lab.			
Experiment No : 8	**Snort Intrusion Detection Package**	**Page**	**1/4**

TITLE :

Use the snort intrusion detection package to analyze traffic and create a signature to identify problem traffic.

OBJECTIVE :

To study snort intrusion detection package to analyze traffic.

THEORY:

Intrusion Detection System (IDS)

An Intrusion Detection System (IDS) is a monitoring system that detects suspicious activities and generates alerts when they are detected. Based upon these alerts, a security operations center (SOC) analyst or incident responder can investigate the issue and take the appropriate actions to remediate the threat.

Intrusion detection systems are designed to be deployed in different environments. And like many cybersecurity solutions, an IDS can either be host-based or network-based.

- **Host-Based IDS (HIDS):** A host-based IDS is deployed on a particular endpoint and designed to protect it against internal and external threats. Such an IDS may have the ability to monitor network traffic to and from the machine, observe running processes, and inspect the system's logs. A host-based IDS's visibility is limited to its host machine, decreasing the available context for decision-making, but has deep visibility into the host computer's internals.
- **Network-Based IDS (NIDS):** A network-based IDS solution is designed to monitor an entire protected network. It has visibility into all traffic flowing through the network and makes determinations based upon packet metadata and contents. This wider viewpoint provides more context and the ability to detect widespread threats; however, these systems lack visibility into the internals of the endpoints that they protect.

Experiment No : 8	Snort Intrusion Detection Package	Page	2/4

SNORT :

SNORT is a network based intrusion detection system which is written in C programming language. It was developed in 1998 by Martin Roesch. Now it is developed by Cisco. It is free open-source software. It can also be used as a packet sniffer to monitor the system in real time. The network admin can use it to watch all the incoming packets and find the ones which are dangerous to the system. It is based on library packet capture tool. The rules are fairly easy to create and implement and it can be deployed in any kind on operating system and any kind of network environment. The main reason of the popularity of this IDS over others is that it is a free-to-use software and also open source because of which any user can able to use it as the way he want.

Features :

- Real-time traffic monitor
- Packet logging
- Analysis of protocol
- Content matching
- OS fingerprinting
- Can be installed in any network environment.
- Creates logs
- Open Source
- Rules are easy to implement

Installation Steps:

In Linux:
Step-1: wget https://www.snort.org/downloads/snort/snort-2.9.15.tar.gz
Step-2: tar xvzf snort-2.9.15.tar.gz
Step-3: cd snort-2.9.15
Step-4: ./configure –enable-sourcefire && make && sudo make install

Basic Usages:

1. **Sniffer Mode :**

   To print TCP/IP header use command **./snort -v**

   To print IP address along with header use command **./snort -vd**

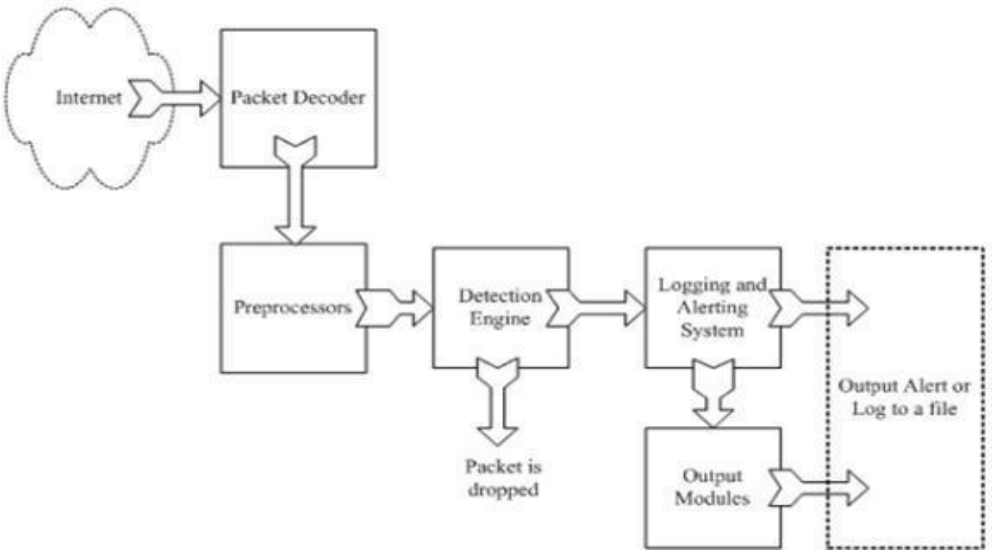
2. **Packet Logging –**

   To store packet in disk you need to give path where you want to store the logs. For this command is**./snort -dev -l ./SnortLogs**.

3. **Activate network intrusion detection mode –**

   To start this mode use this command **./snort -dev -l ./SnortLogs -h 192.127.1.0/24 -c snort.conf**

Components of Snort based Intrusion Detection System



Computer Network & Security Lab.			
Experiment No : 8	**Snort Intrusion Detection Package**	**Page**	**4/4**

- **A Packet Decoder:** It takes packets from different networks and prepares them for preprocessing or any further action. It basically decodes the coming network packets.
- **A Preprocessor**: It prepares and modifies the data packets and also performs defragmentation of data packets, decodes the TCP streams.
- **A Detection Engine**: It performs packet detection on the basis of Snort rules. If any packet matches the rules, appropriate action is taken, else it is dropped.
- **Logging and Alerting System**: The detected packet is either logged in system files or in case of threats, the system is alerted.
- **Output Modules**: They control the type of output from the logging and alert system.

Advantages of Intrusion Detection Systems

- The network or computer is constantly monitored for any invasion or attack.
- The system can be modified and changed according to the needs of specific clients and can help outside as well as inner threats to the system and network.
- It effectively prevents any damage to the network.
- It provides a user-friendly interface which allows easy security management systems.
- Any alterations to files and directories on the system can be easily detected and reported.

CONCLUSION :

Thus used the snort intrusion detection package to analyse traffic .