



Experiment - 1

Information Security Lab

Aim

Make an experiment to implement WEP/WPA 2 PSK, 802.1x EAP Security Protocol.

Syeda Reeha Quasar

14114802719

7C7

EXPERIMENT – 1

Aim:

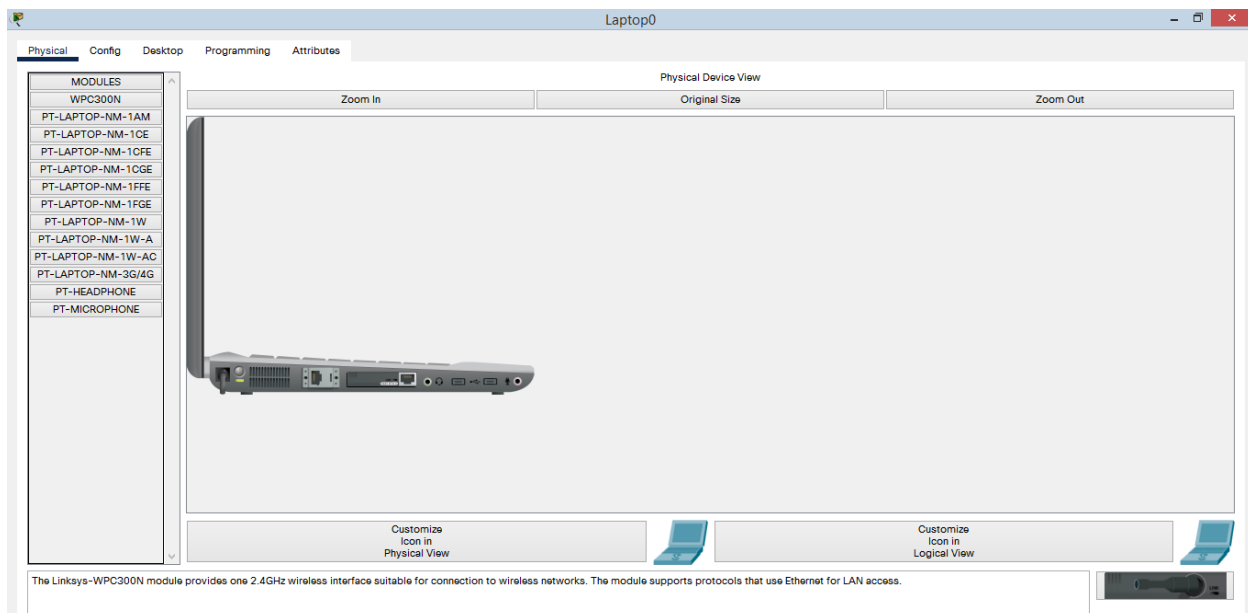
Make an experiment to implement WEP/WPA 2 PSK, 802.1x EAP Security Protocol.

Theory:

Cisco Packet Tracer is an innovative network simulation and visualization tool. This free software helps you to practice your network configuration and troubleshooting skills via your desktop computer or an Android or iOS based mobile device. Packet Tracer is available for both the Linux and Windows desktop environments.

With Packet Tracer you can choose to build a network from scratch, use a pre-built sample network, or complete classroom lab assignments. Packet Tracer allows you to easily explore how data traverses your network. Packet Tracer provides an easy way to design and build networks of varying sizes without expensive lab equipment. While this software is not a replacement for practicing on physical routers, switches, firewalls, and such.

Physical Device View



IOS Command Line Interface

Physical Config CLI Attributes

IOS Command Line Interface

```

--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: yes

At any point you may enter a question mark '?' for help.
Use ctrl-c to abort configuration dialog at any prompt.
Default settings are in square brackets '[]'.

Basic management setup configures only enough connectivity
for management of the system, extended setup will ask you
to configure each interface on the system

Would you like to enter basic management setup? [yes/no]: yes
Configuring global parameters:

Enter host name [Router]: Router

The enable secret is a password used to protect access to
privileged EXEC and configuration modes. This password, after
entered, becomes encrypted in the configuration.
Enter enable secret: Ayush5Pandey#

The enable password is used when you do not specify an
enable secret password, with some older software versions, and
some boot images.
Enter enable password: Ayush5Pandey#
Please choose a password that is different from the enable secret
Enter enable password: Ayush5Pandey##

The virtual terminal password is used to protect
access to the router over a network interface.
Enter virtual terminal password: Ayush_Pandey
Configure SNMP Network Management? [no]:no

Current interface summary

Interface          IP-Address      OK? Method Status          Protocol

```

Ctrl+F6 to exit CLI focus

Copy Paste

Creating the Profile

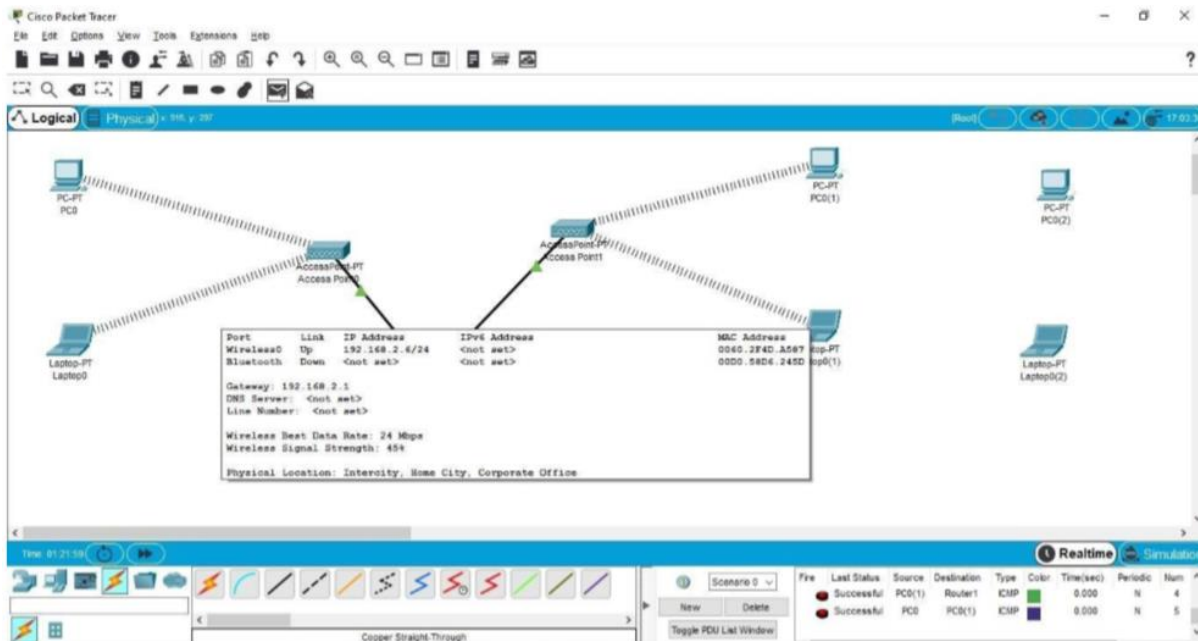
The screenshot shows the Cisco Packet Tracer interface with a 'Creating a Profile' dialog box open. The dialog has tabs for Physical, Config, Desktop, Programming, and Attributes. The 'Desktop' tab is active, showing the 'Creating a Profile' window. Inside this window, there's a section titled 'Available Wireless Networks' with instructions to select a network and click 'Connect'. Below this is a table with columns: Wireless Network Name, CH, Signal, and Security. The table lists two networks: DISCO and DISCOB, both with CH 1 and Signal 38%. There are 'Refresh' and 'Connect' buttons. At the bottom, there's a 'Wireless-N Notebook Adapter' and a 'Wireless Network Monitor v1.0' (Model No. WPC300N). The background shows a network diagram with a PC-PT PC0 connected to an Access Point, and a Laptop-PT Laptop0 connected to the same Access Point. The bottom status bar shows 'Realtime' simulation mode and a packet capture table with columns: Pkts, Last Status, Source, Destination, Type, Color, Time(sec), Periodic, Num. The packet capture table shows two packets: one successful from PC0 to PC0(1) and one failed from Laptop0 to Laptop0, both of type ICMP.

| Wireless Network Name | CH | Signal | Security |
|-----------------------|----|--------|----------|
| DISCO | 1 | 38% | |
| DISCOB | 1 | 38% | |

| Pkts | Last Status | Source | Destination | Type | Color | Time(sec) | Periodic | Num |
|------|-------------|----------|-------------|------|-------|-----------|----------|-----|
| 5 | Successful | PC0 | PC0(1) | ICMP | | 0.000 | N | 5 |
| 6 | Failed | Lapto... | Laptop0 | ICMP | | 0.000 | N | 6 |

Making Configurations



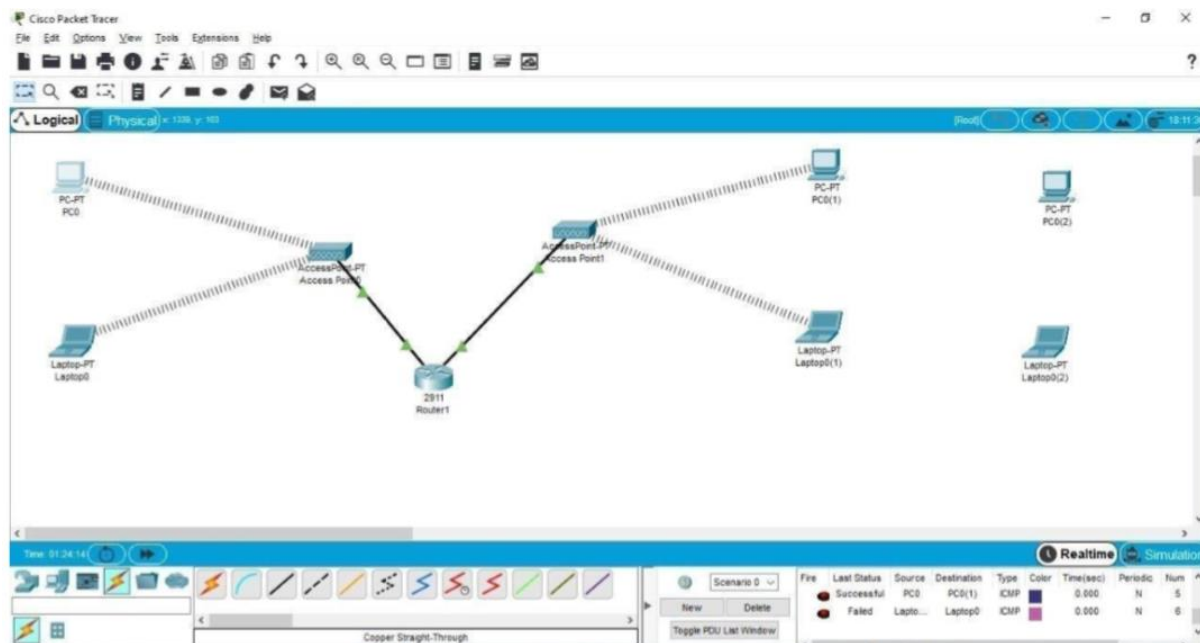


WEP (Wired Equivalent Privacy)

- It is a Security algorithm for wireless network.
- It is designed to provide Wireless Local Area Network (WLAN) with a level of security and privacy comparable to what is usually expected of a wired LAN.
- To provide data confidentiality comparable to that of a traditional wired network.
- A 64-bit WEP key is usually entered as a string of 10 hexadecimal (base 16) characters (0–9 and A–F). Each character represents 4 bits, 10 digits of 4 bits each gives 40 bits; adding the 24-bit IV produces the complete 64bit WEP key (4 bits × 10 + 24 bits IV = 64 bits of WEP key).

WEP is a security algorithm for IEEE 802.11 wireless networks, ratified in 1997, its intention was to provide data confidentiality comparable to that of a traditional wired network. WEP, recognizable by its key of 10 or 26 hexadecimal digits (40 or 104 bits), was at one time widely in use and was often the first security choice presented to users by router configuration tools.

Setting Up the WPA2 Network



WPA stands for Wi-Fi Protected Access is a security standard for users of computing devices equipped with wireless internet connections. WPA was developed by the Wi-Fi Alliance to provide more sophisticated data encryption and better user authentication than Wired Equivalent Privacy (WEP), the original Wi-Fi security standard. The new standard, which was ratified by the IEEE in 2004 as 802.11i, was designed to be backward-compatible with WEP to encourage quick, easy adoption. Network security professionals were able to support WPA on many WEP-based devices with a simple firmware update.

Result: The working of WEP and WPA2 was successfully implemented and understood.



Experiment - 2

Information Security Lab

Aim

To implement Windows Firewall.

Syeda Reeha Quasar

14114802719

7C7

EXPERIMENT – 2

Aim:

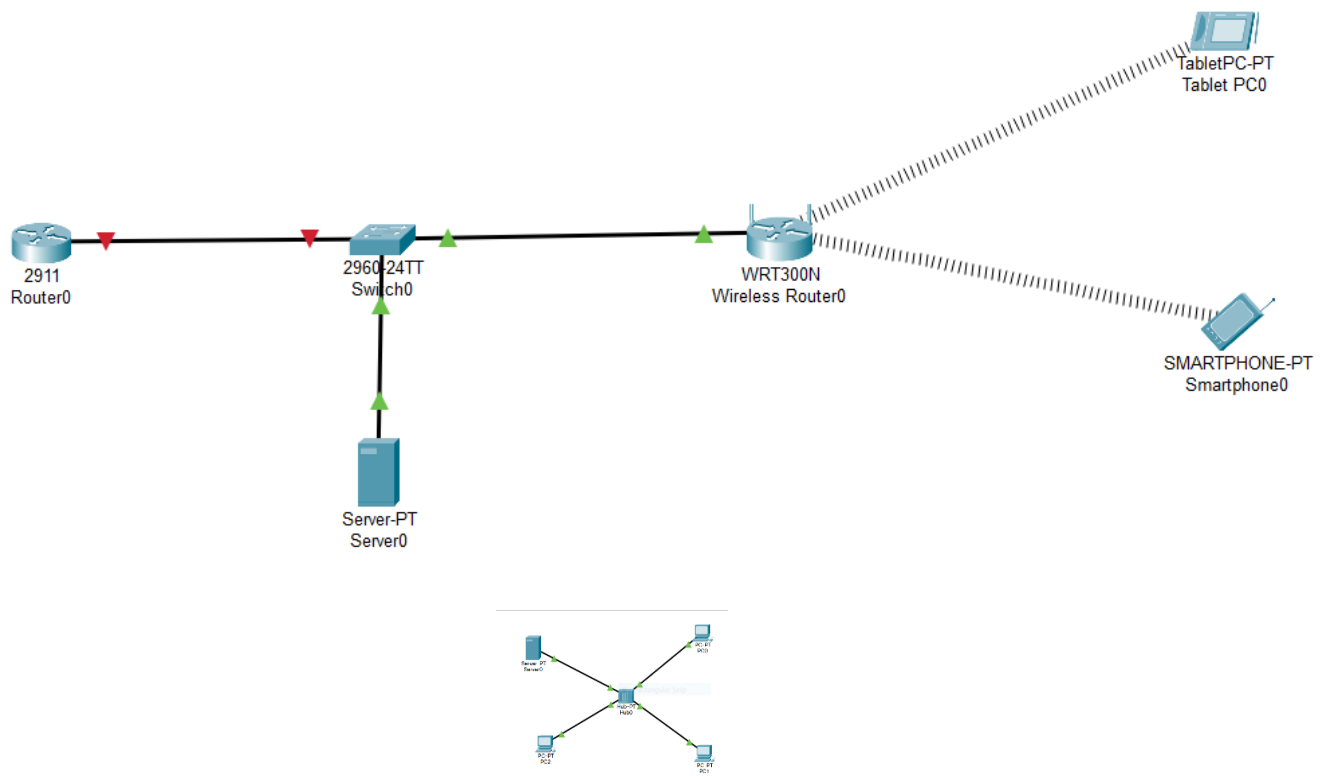
Implement firewall through App to login into bank-site,; to implement E-commerce, debit card transaction through payment gateway.

Theory:

A firewall is a network security device that monitors incoming and outgoing network traffic and permits or blocks data packets based on a set of security rules. Its purpose is to establish a barrier between your internal network and incoming traffic from external sources (such as the internet) in order to block malicious traffic like viruses and hackers.

Firewalls carefully analyze incoming traffic based on pre-established rules and filter traffic coming from unsecured or suspicious sources to prevent attacks. Firewalls guard traffic at a computer's entry point, called ports, which is where information is exchanged with external devices. For example, "Source address 172.18.1.1 is allowed to reach destination 172.18.2.1 over port 22."

Screenshots:



Physical **Config** Services Desktop Programming Attributes

GLOBAL

Settings

Algorithm Settings

INTERFACE

FastEthernet0

FastEthernet0

Port Status ☒ On

Bandwidth ☐ 100 Mbps ☐ 10 Mbps ☒ Auto

Duplex ☐ Half Duplex ☐ Full Duplex ☒ Auto

MAC Address 0001.4366.B167

IP Configuration

☐ DHCP

☒ Static

IPv4 Address 20.0.0.1

Subnet Mask 255.0.0.0

IPv6 Configuration

☐ Automatic

☒ Static

IPv6 Address

Link Local Address: FE80::201:43FF:FE66:B167

Server0

Physical Config **Services** Desktop Programming Attributes

SERVICES

HTTP

DHCP

DHCPv6

TFTP

DNS

SYSLOG

AAA

NTP

EMATI

HTTP

☒ On ☐ Off

HTTPS

☒ On ☐ Off

File Manager

| | File Name | Edit | Delete |
|---|-----------------------|--------|----------|
| 1 | copyrights.html | (edit) | (delete) |
| 2 | cscoptloqo177X111.ipq | | (delete) |

Server0

Physical Config Services **Desktop** Programming Attributes

Firewall

Service ☒ On ☐ Off

Interface FastEthernet0

Inbound Rules

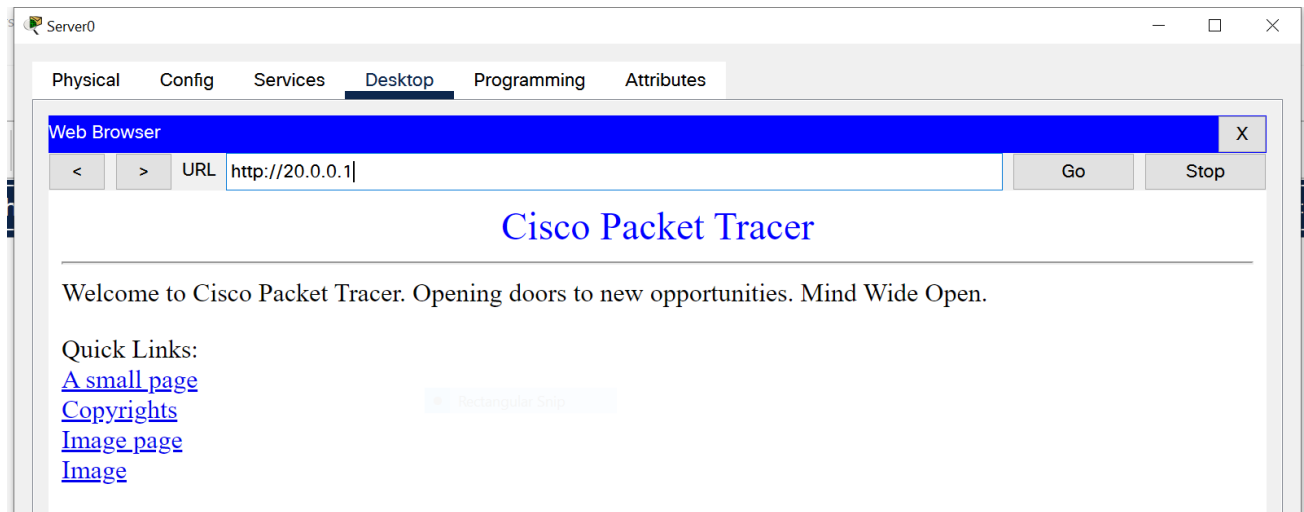
Action Protocol

Remote IP Remote Wildcard Mask

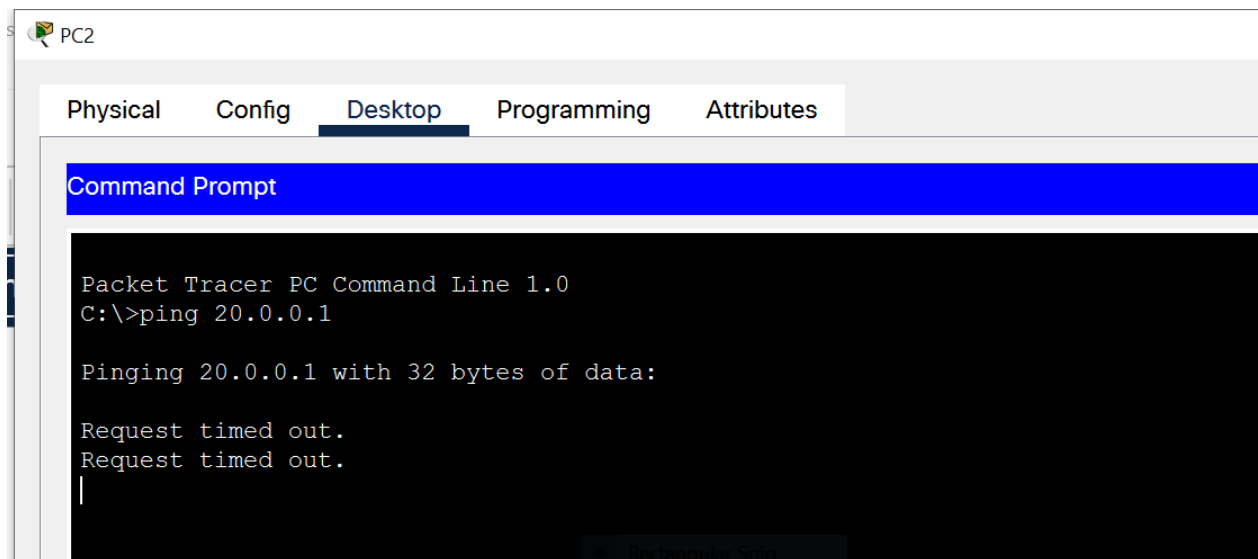
Remote Port Local Port

Save Remove Add

| | Action | Protocol | Remote IP | Remote Wild Card | Remote Port | Local Port |
|---|--------|----------|-----------|------------------|-------------|------------|
| 1 | Allow | IP | 0.0.0.0 | 255.255.255.255 | - | - |
| 2 | Deny | ICMP | 0.0.0.0 | 255.255.255.255 | - | - |



Positive Response in Browser



Request Timed out on PC2.

Result: Working of a Firewall is Understood and Implemented.



Experiment - 3

Information Security Lab

Aim

Implement bio-metric system to have physical security through different access control permissions.

Syeda Reeha Quasar

14114802719

7C7

EXPERIMENT – 3

Aim:

Implement bio-metric system to have physical security through different access control permissions.

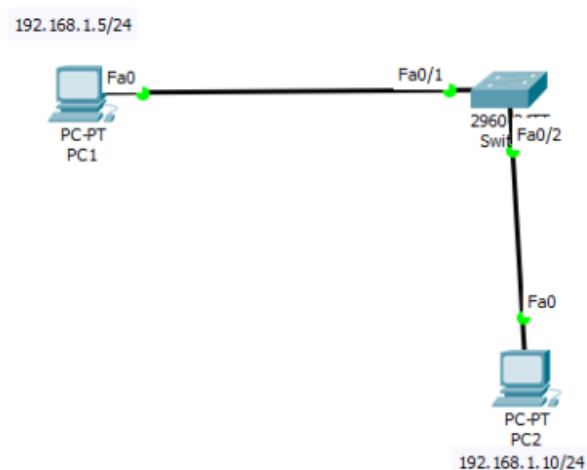
Theory:

Switch port Security is a **network security** feature that associates specific MAC addresses of devices (such as PCs) with specific interfaces on a switch. This will enable you to restrict access to a given switch interface so that only the authorized devices can use it. If an unauthorized device is connected to the same port, you can define the action that the switch will take, such as discarding the traffic, sending an alert, or shutting down the port.



Now let's configure port security in Packet Tracer.

1. Build the network topology:



PC1 connects to fa0/1 and PC2 to fa0/2 of the switch

2. Now configure switch port security on switch interfaces.

We'll configure port security interfaces on fa0/1 and fa0/2. To do this, we'll:

- Configure the port as an **access port**
- Enable **port security**
- Define which **MAC addresses** are allowed to send frames through this interface.

Here are the commands:

```
Switch(config)#int fa0/1
```

```
Switch(config-if)#switchport mode access
```

```
Switch(config-if)#switchport port-security
```

```
Switch(config-if)#switchport port-security mac-address sticky
```

The *sticky* keyword instructs the switch to **dynamically** learn the MAC address of the currently connected host.

You can add these two optional commands.

- defining the action that the switch will take when a frame from an unauthorized device is received. This is done using the switchport port-security violation {protect | restrict | shutdown} interface command. All three options discard the traffic from the unauthorized device.
- defining the maximum number of MAC addresses that can be received on the port using the switchport port-security maximum NUMBER interface submode command

Let's add the above 2 commands to our configuration:

```
Switch(config-if)#switchport port-security violation shutdown
```

```
Switch(config-if)#switchport port-security maximum 1
```

We're done with port security configuration for fa0/1

Moving on...

In a similar way to switch interface *fa0/1*, configure switch port security for *fa0/2* connected to PC2:

```
Switch(config)#interface fa0/2
```

```
Switch(config-if)#switchport mode access
```

```
Switch(config-if)#switchport port-security
```

```
Switch(config-if)#switchport port-security mac-address sticky
```

```
Switch(config-if)#switchport port-security violation shutdown
```

```
Switch(config-if)#switchport port-security maximum 1
```

That's all for port-security configuration on fa0/2

A shorthand method for configuration: The port security configurations for both fa0/1 and fa0/2 could be done more faster with the help of interface range command as shown below:

```
Switch(config-if-range)#interface range fa0/1-2
```

```
Switch(config-if)#switchport mode access
```

```
Switch(config-if)#switchport port-security
```

```
Switch(config-if)#switchport port-security mac-address sticky
```

```
Switch(config-if)#switchport port-security violation shutdown
```

```
Switch(config-if)#switchport port-security maximum 1
```

Here, we define a range of interfaces on which we want to configure port security, then proceed to configure port-security for all the interfaces specified at a go instead of one interface at a time.

The interface range command can save you tons of work in doing individual configurations if you were configuring port security for many switch interfaces, say, 24 ports on a switch.

Next,

4. We'll verify port security configurations on interfaces fa0/1 and fa0/2

To verify if the switch has learnt the MAC address of PC1, you can use the command:

show port-security interface fa0/1

```
Switch#  
Switch#show port-security interface fa0/1  
Port Security           : Enabled  
Port Status             : Secure-up  
Violation Mode          : Shutdown  
Aging Time              : 0 mins  
Aging Type              : Absolute  
SecureStatic Address Aging : Disabled  
Maximum MAC Addresses   : 1  
Total MAC Addresses     : 1  
Configured MAC Addresses : 0  
Sticky MAC Addresses    : 1  
Last Source Address:Vlan : 0005.5E57.A982:1  
Security Violation Count : 0
```

Verify that the switch has learnt the MAC address of PC1.

You may also use the command: show port-security address

```
Switch#show port-security address
```

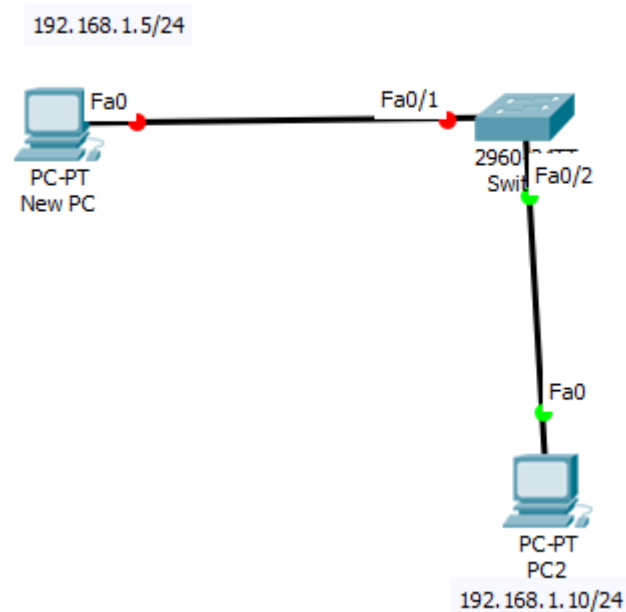
| Secure Mac Address Table | | | |
|--------------------------|-----------------|--------------|-------|
| Vlan | Mac Address | Type | Ports |
| | Remaining Age | | |
| | (mins) | | |
| 1 | 0005.5E57.A982 | SecureSticky | |
| | FastEthernet0/1 | - | |
| 1 | 0007.EC08.B84C | SecureSticky | |
| | FastEthernet0/2 | - | |

```
Total Addresses in System (excluding one mac per port) : 0
Max Addresses limit in System (excluding one mac per port) : 1024
```

Try also pinging PC2 from PC1. Ping should be successful here since switch port security is not violated.

The case of Port Security Violation

Now connect a different PC to fa0/1 in place of PC1. See the effect of doing this:



Notice that fa0/1 shuts down upon connecting the new PC, as indicated by the red LED.

This is because the switch had already associated fa0/1 with the MAC address of PC1 and the **maximum** number of MAC addresses that we defined for this port is **1**. So attaching the new PC to fa0/1 violates the port security rules that we set and as a result, the interface **shuts down**.

You can verify this further by using the command we used before: show port-security interface fa0/1

```
Switch#  
Switch#show port-security interface fa0/1  
Port Security           : Enabled  
Port Status             : Secure-shutdown  
Violation Mode          : Shutdown  
Aging Time              : 0 mins  
Aging Type              : Absolute  
SecureStatic Address Aging : Disabled  
Maximum MAC Addresses   : 1  
Total MAC Addresses     : 1  
Configured MAC Addresses : 0  
Sticky MAC Addresses    : 1  
Last Source Address:Vlan : 0001.64C3.8971:1  
Security Violation Count : 1
```

Verify from above that **port status** is now **Secure-shutdown** upon violation of port security.

Further, a ping from the New PC to PC2 will definitely fail because the switch cannot forward a frame via an interface that is shut down.

How to Reset an interface that has been shut down due to Violation of Port Security:

One of the options on the table is to manually restart the shutdown interface(fa0/1 in our case here). Unplug the cable from unauthorized PC(new PC) and plug it back to authorized PC(PC1)

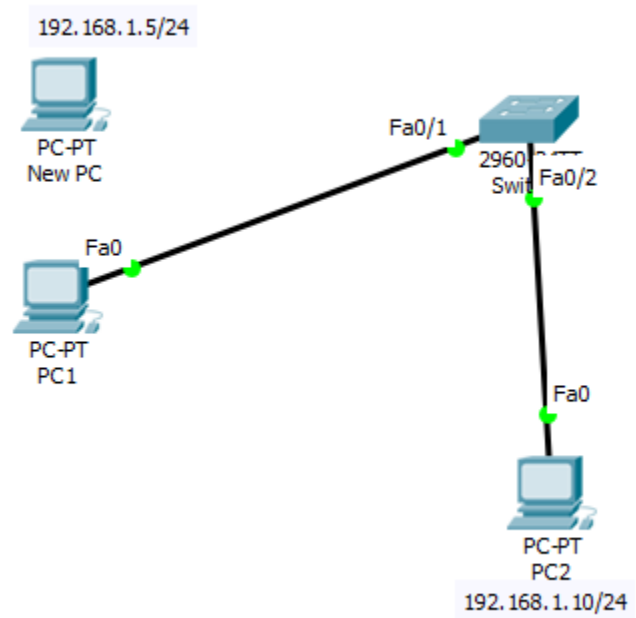
Then run following commands on switch and test connectivity from the authorized PC (PC1):

```
Switch(config)#interface fa0/1
```

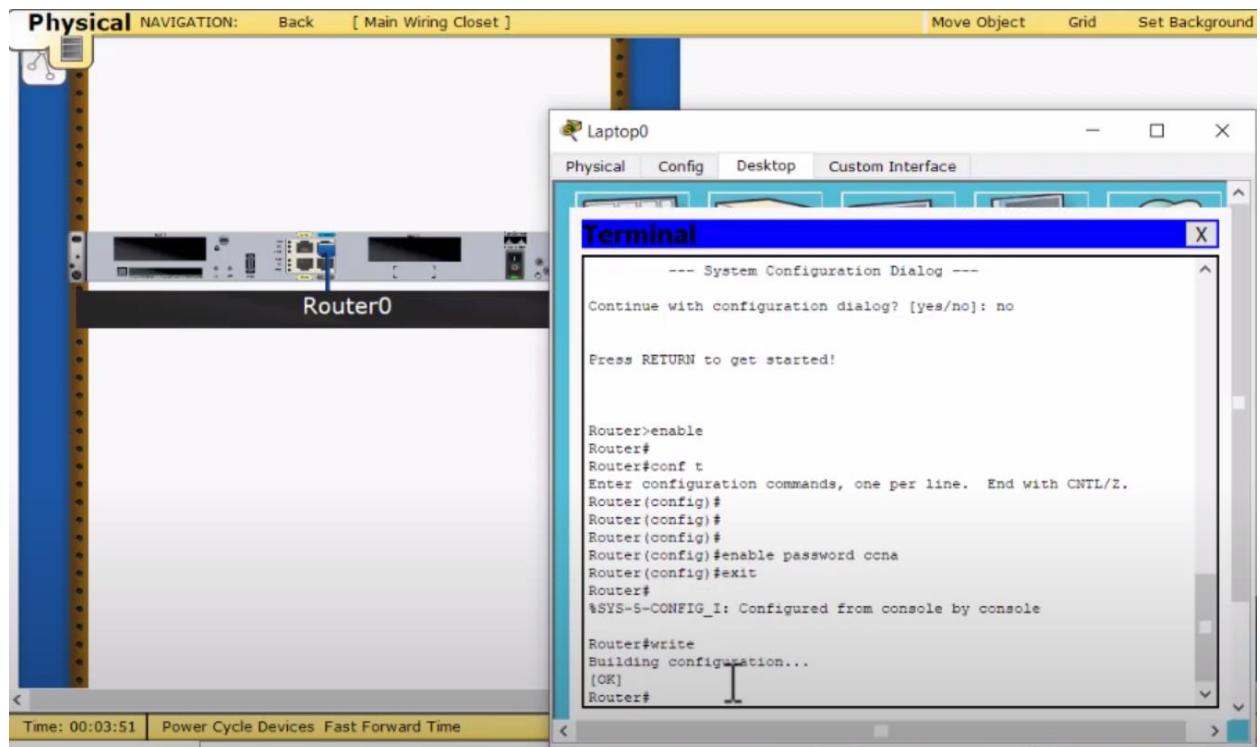
```
Switch(config-if)#shutdown
```

```
Switch(config-if)#no shutdown
```

Now the interface fa0/1 should change status to up.



Success! Success!



Tool Used: Burp Intruder

Target Website: <http://altoromutual.com> provided by IBM for pen testing the applications.

In this we would use the Burp tool to modify the connection in order to access the account of another user. We will setup the application and hook the browser with the burp tool using the proxy so that all the information communicated is done by the burp tool and you can modify any communication that takes place between the two.

Sample Instructions

Login: jsmith

Password: Demo1234

Welcome page after login

The screenshot shows the Altoro Mutual Online Banking Login page. The browser address bar displays "altoromutual.com:8080/login.jsp". The page features the Altoro Mutual logo at the top. Below the logo, there is a navigation bar with "PERSONAL" and "SMALL BUSINESS" tabs. The "PERSONAL" tab is selected. The main content area is titled "Online Banking Login" and contains a login form with fields for "Username" (containing "jsmith") and "Password" (containing "Demo1234"), and a "Login" button. A sidebar on the left lists various services under "PERSONAL", "SMALL BUSINESS", and "INSIDE ALTORO MUTUAL".

The screenshot shows the Altoro Mutual My Account page. The browser address bar displays "altoromutual.com:8080/bank/main.jsp". The page features the Altoro Mutual logo at the top. Below the logo, there is a navigation bar with "MY ACCOUNT", "PERSONAL", and "SMALL BUSINESS" tabs. The "MY ACCOUNT" tab is selected. The main content area is titled "Hello John Smith" and contains a welcome message, a "View Account Details" section with a dropdown menu showing "800002 Savings" and a "GO" button, and a "Congratulations!" message stating "You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!". A sidebar on the left lists various services under "I WANT TO ...".

The Altoro website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental to your use of this website. For more information, please go to <http://www-142.ibm.com/software/products/usen/subcategorySW10>.

Copyright © 2008, 2022, IBM Corporation, All rights reserved.

Altoro Mutual

Not secure | altoromutual.com:8080/bank/transaction.jsp

See Off | Control Us | Feedback | Search [GO]

Altoro Mutual

MY ACCOUNT PERSONAL SMALL BUSINESS ROIDE ALTORO MUTUAL

I WANT TO ...
[View Account Summary](#)
[View Recent Transactions](#)
[Transfer Funds](#)
[Search News Articles](#)
[Customize Site Language](#)

Recent Transactions

After Before Submit

2009-mm-dd 2009-mm-dd

| Transaction ID | Transaction Time | Account ID | Action | Amount |
|----------------|------------------|------------------|------------------|-----------|
| 2285 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2284 | 2009-11-24 09:35 | 000003 | Withdrawal | -\$1.00 |
| 2283 | 2009-11-24 09:35 | 423906C020596288 | Cash Advance Fee | \$2.50 |
| 2280 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2249 | 2009-11-24 09:35 | 423906C020596288 | Cash Advance | \$1.00 |
| 2262 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2281 | 2009-11-24 09:35 | 000002 | Withdrawal | -\$1.00 |
| 2248 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2247 | 2009-11-24 09:35 | 000002 | Withdrawal | -\$1.00 |
| 2246 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2245 | 2009-11-24 09:35 | 000003 | Withdrawal | -\$1.00 |
| 2244 | 2009-11-24 09:35 | 423906C020596288 | Cash Advance Fee | \$2.50 |
| 2243 | 2009-11-24 09:35 | 000002 | Deposit | \$1.00 |
| 2242 | 2009-11-24 09:35 | 423906C020596288 | Cash Advance | \$1.00 |
| 2241 | 2009-11-24 09:32 | 423906C020596288 | Payment | \$1.00 |
| 2240 | 2009-11-24 09:32 | 000002 | Withdrawal | -\$1.00 |
| 2239 | 2009-11-24 09:32 | 423906C020596288 | Cash Advance Fee | \$2.50 |
| 2238 | 2009-11-24 09:32 | 000002 | Deposit | \$1.00 |
| 2237 | 2009-11-24 09:32 | 423906C020596288 | Cash Advance | \$1.00 |
| 2236 | 2009-11-24 09:32 | 000002 | Deposit | \$1.00 |
| 2235 | 2009-11-24 09:32 | 000003 | Withdrawal | -\$1.00 |
| 2234 | 2009-11-24 09:32 | 000003 | Deposit | \$1.00 |
| 2233 | 2009-11-24 09:32 | 000002 | Withdrawal | -\$1.00 |
| 2232 | 2009-11-24 09:32 | 000002 | Deposit | \$1.00 |
| 2231 | 2009-11-24 09:32 | 000002 | Withdrawal | -\$1.00 |
| 2222 | 2019-03-11 16:00 | 000003 | Deposit | \$400.00 |
| 2221 | 2019-03-11 16:00 | 000002 | Withdrawal | -\$400.00 |
| 2220 | 2019-03-08 09:22 | 000003 | Deposit | \$100.00 |

The screenshot displays the Burp Suite interface. At the top, the title bar reads "Burp Suite Community Edition v2022.12.5 - Temporary Project". The main menu includes Burp, Project, Intruder, Repeater, Window, and Help. Below the menu is a toolbar with tabs for Dashboard, Target, Proxy, Intruder, Repeater, Sequencer, Decoder, Comparer, Logger, Extensions, and Learn. The Proxy tab is active, showing a list of intercepted requests. The first request is selected, showing its details in the Raw view. The request is a GET to /bank/transaction.jsp. The request body is visible, containing a cookie with a JWT token and a header with a Sec-Fetch-Site: same-origin. The Inspector panel on the right shows the selected text: ODAwMDAzf1NhdmLUZ3M%3d.

Altoro Mutual

PERSONAL SMALL BUSINESS INSIDE ALTORO MUTUAL

Recent Transactions

Alter Before

| Transaction ID | Transaction Time | Account ID | Action | Amount |
|----------------|------------------|------------------|------------|------------|
| 5453 | 2022-12-27 08:05 | 4630002130206208 | Payment | \$100.00 |
| 5459 | 2022-12-27 08:05 | 800003 | Withdrawal | -\$100.00 |
| 5458 | 2022-12-27 07:58 | 800003 | Deposit | \$1234.00 |
| 5457 | 2022-12-27 07:58 | 800003 | Withdrawal | -\$1234.00 |
| 5456 | 2022-12-27 07:58 | 800003 | Deposit | \$1234.00 |
| 5455 | 2022-12-27 07:58 | 800003 | Withdrawal | -\$1234.00 |
| 5454 | 2022-12-27 07:58 | 800003 | Deposit | \$1234.00 |
| 5453 | 2022-12-27 07:58 | 800003 | Withdrawal | -\$1234.00 |
| 5452 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5451 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5450 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5389 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5388 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5387 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5386 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5385 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5384 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5383 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5382 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5381 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5380 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |
| 5379 | 2022-12-27 07:57 | 800003 | Withdrawal | -\$1234.00 |
| 5378 | 2022-12-27 07:57 | 800003 | Deposit | \$1234.00 |

As soon as we modify and press the forward button we are being shown the information of another user on whom we have not even logged in. here we are successfully able to present the vulnerability for the broken authentication and session management on the website.

Result: Security techniques are implemented and understood.



Experiment - 4

Information Security Lab

Aim

. Implement RSA algorithm.

Syeda Reeha Quasar

14114802719

7C7



Experiment - 5

Information Security Lab

Aim

Implement DES algorithm.

Syeda Reeha Quasar

14114802719

7C7



Experiment - 6

Information Security Lab

Aim

Implement Diffie-Hellman algorithm.

Syeda Reeha Quasar

14114802719

7C7



Experiment - 7

Information Security Lab

Aim

. Make a study of anyone simulation tool based on parameters of information security.

Syeda Reeha Quasar

14114802719

7C7



Experiment - 8

Information Security Lab

Aim

Implement VPN through Packet-Tracer or any other network simulator tool.

Syeda Reeha Quasar

14114802719

7C7

EXPERIMENT – 4

Aim:

To perform RSA Algorithm.

Theory:

RSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes that the Public Key is given to everyone and Private key is kept private. 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. It is also one of the oldest. The acronym RSA comes from the surnames of Ron Rivest, Adi Shamir and Leonard Adleman, who publicly described the algorithm in 1977.

An example of asymmetric cryptography:

1. A client (for example browser) sends its public key to the server and requests for some data.
2. The server encrypts the data using client's public key and sends the encrypted data.
3. Client receives this data and decrypts it.

Since this is asymmetric, nobody else except browser can decrypt the data even if a third party has public key of browser. The idea of RSA is based on the fact that it is difficult to factorize a large integer. The public key consists of two numbers where one number is multiplication of two large prime numbers. And private key is also derived from the same two prime numbers. So, if somebody can factorize the large number, the private key is compromised. Therefore, encryption strength totally lies on the key size and if we double or triple the key size, the strength of encryption increases exponentially. RSA keys can be typically 1024 or 2048 bits long, but experts believe that 1024-bit keys could be broken in the near future. But till now it seems to be an infeasible task.

Why RSA Encryption is secure?

The idea of making one of your own encryption algorithms public on the internet seems very strange at first. However, this is actually one of the most important steps in RSA encryption.

If Person C intercepts your message to Person B, they already know the encryption key (exponent e , modulus n). However, what he/she doesn't have is the decryption exponent d . Since you encrypted your message with Person B's encryption key, only Person B has the decryption key (exponent d , modulus n) to decrypt it. Person C is only missing one piece of information, exponent d , which turns out to be the hardest piece of information to find.

Person C also knows that

$$de \equiv 1 \pmod{\phi(n)}, \text{ or } de \equiv 1 \pmod{(p-1)(q-1)}.$$

Since he/she knows that $n = pq$, the simplest way to find n would be to somehow factor n into the exact primes used by Person B in the algorithm. From there, he/she could simply calculate the congruence to find d .

With larger (which are more secure) primes, this turns out to be nearly impossible to do.

If $p = 7717$ and $q = 7919$, n would be 61110923. If we let $e = 5$, then all Person

C knows is

$$e = 5, n = 61110923.$$

Clearly, it would take very long to factor n , but imagine what would happen if

$$p = 982451653, q = 961748941.$$

Then n would be 944871836856449473.

Now factoring n is basically impossible to do by hand. However, even this value of n is smaller than most values of n used in RSA Encryption. It took 290 computers over the internet and a supercomputer 4 months to find that

$$\begin{aligned} n = &10941738641570527421809707322040357612003732945449205990913842131476349984 \\ &288934784717997257891267332497625752899781833797076537244027146743 \\ &531593354333897 \end{aligned}$$

Had prime factors

$$p = 102639592829741105772054196573991675900716567808038066803341933521790711307779,$$

$$q = 106603488380168454820927220360012878679207958575989291522270608237193062808643.$$

In this case n had only 155 digits. Many values of n have over 200 digits, making the RSA algorithm nearly unbreakable.

Algorithm:

Generating Public Key:

1. Select two prime numbers
 - a. Suppose $P = 53$ and $Q = 59$.
2. Now First part of the Public key: $n = P * Q = 3127$.
3. We also need a small exponent say e
4. We need to calculate $\Phi(n)$:
 - a. Such that $\Phi(n) = (P-1) (Q-1)$
 - b. so, $\Phi(n) = 3016$
5. Now calculate Private Key, d :
 - a. $d = (k * \Phi(n) + 1) / e$ for some integer k
 - b. For $k = 2$, value of d is 2011.
6. Now we are ready with our –
Public Key ($n = 3127$ and $e = 3$) and Private Key ($d = 2011$)

Now we will encrypt "HI":

Convert letters to numbers: $H = 8$ and $I = 9$

Thus, Encrypted Data $c = 89e \bmod n$.

Thus, our Encrypted Data comes out to be 1394

Now we will decrypt 1349:

Decrypted Data = $cd \bmod n$.

Thus, our Encrypted Data comes out to be 89

$8 = H$ and $I = 9$ i.e. "HI".

Source Code:

```
#include <bits/stdc++.h>
using namespace std;

set<int> prime;
```

```

int public_key;
int private_key;
int n;

void primefiller()
{
    vector<bool> seive(250, true);
    seive[0] = false;
    seive[1] = false;
    for (int i = 2; i < 250; i++)
    {
        for (int j = i * 2; j < 250; j += i)
        {
            seive[j] = false;
        }
    }
    for (int i = 0; i < seive.size(); i++)
    {
        if (seive[i])
            prime.insert(i);
    }
}

int pickrandomprime()
{
    int k = rand() % prime.size();
    auto it = prime.begin();
    while (k--)
        it++;
    int ret = *it;
    prime.erase(it);
    return ret;
}

void setkeys()
{
    int prime1 = pickrandomprime();
    int prime2 = pickrandomprime();

    n = prime1 * prime2;
    int fi = (prime1 - 1) * (prime2 - 1);
    int e = 2;
    while (1)
    {
        if (__gcd(e, fi) == 1)

```

```

        break;
    e++;
}
public_key = e;
int d = 2;
while (1)
{
    if ((d * e) % fi == 1)
        break;
    d++;
}
private_key = d;
}

long long int encrypt(double message)
{
    int e = public_key;
    long long int encrpyted_text = 1;
    while (e--)
    {
        encrpyted_text *= message;
        encrpyted_text %= n;
    }
    return encrpyted_text;
}

long long int decrypt(int encrpyted_text)
{
    int d = private_key;
    long long int decrypted = 1;
    while (d--)
    {
        decrypted *= encrpyted_text;
        decrypted %= n;
    }
    return decrypted;
}

vector<int> encoder(string message)
{
    vector<int> form;

    for (auto &letter : message)
        form.push_back(encrypt((int)letter));
    return form;
}

```

```

}
string decoder(vector<int> encoded)
{
    string s;

    for (auto &num : encoded)
        s += decrypt(num);
    return s;
}

int main()
{
    primefiller();
    setkeys();

    string message = "Syeda Reeha Quasar 14114802719";

    vector<int> coded = encoder(message);
    cout << "Name and Roll:\n"
         << message;
    cout << "\n\nThe encoded message(encrypted by public key)\n";
    for (auto &p : coded)
        cout << p;
    cout << "\n\nThe decoded message(decrypted by private key)\n";
    cout << decoder(coded) << endl;

    return 0;
}

```

Output:

```

PS E:\sem 7\IS\codes> & .\"rsa.exe"
Name and Roll:
Syeda Reeha Quasar 14114802719

The encoded message(encrypted by public key)
4253427142313213631568310614214233481363156815742016136334211363134915684922834949228330318583356328493154

The decoded message(decrypted by private key)
Syeda Reeha Quasar 14114802719
PS E:\sem 7\IS\codes> 

```

Result: RSA was implemented and understood.

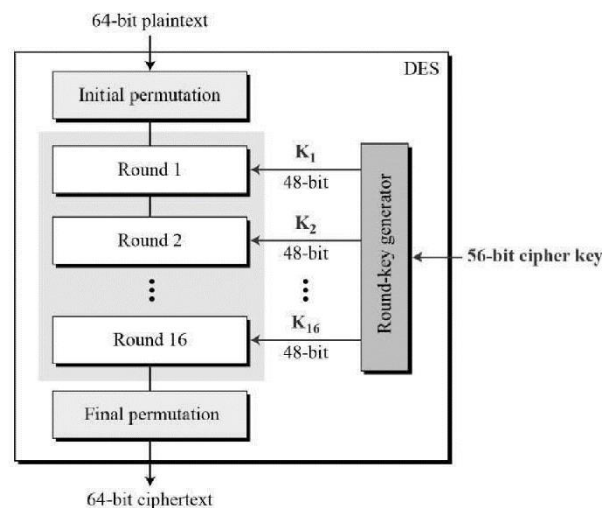
EXPERIMENT – 5

Aim:

To implement DES algorithm.

Theory:

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST). DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm.



Since DES is based on the Feistel Cipher, all that is required to specify DES is –

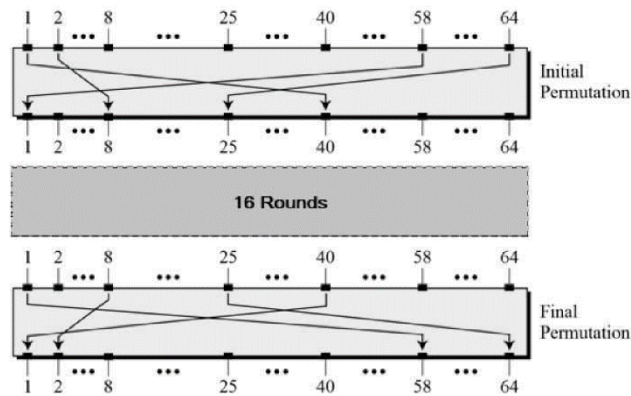
Round function

Key schedule

Any additional processing – Initial and final permutation

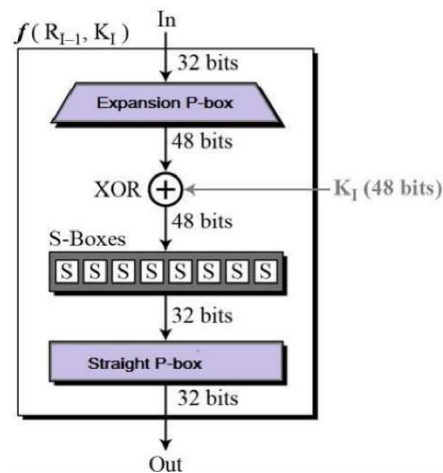
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows –

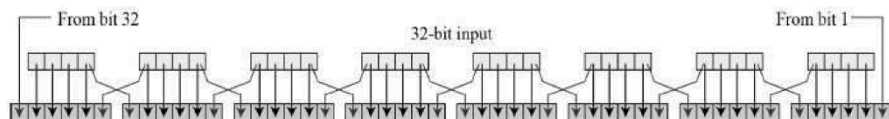


Round Function

The heart of this cipher is the DES function, f . The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



Expansion Permutation Box – Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration –

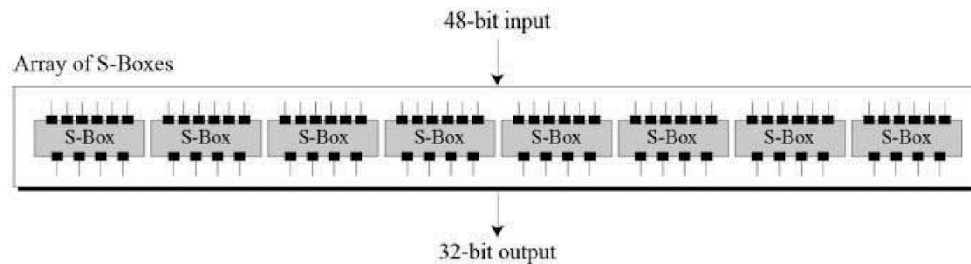


The graphically depicted permutation logic is generally described as table in DES specification illustrated as shown –

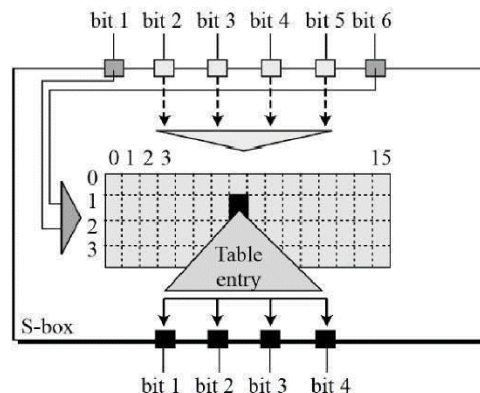
| | | | | | |
|----|----|----|----|----|----|
| 32 | 01 | 02 | 03 | 04 | 05 |
| 04 | 05 | 06 | 07 | 08 | 09 |
| 08 | 09 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 31 | 31 | 32 | 01 |

XOR (Whitener). – After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.

Substitution Boxes. – The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration –



The S-box rule is illustrated below –



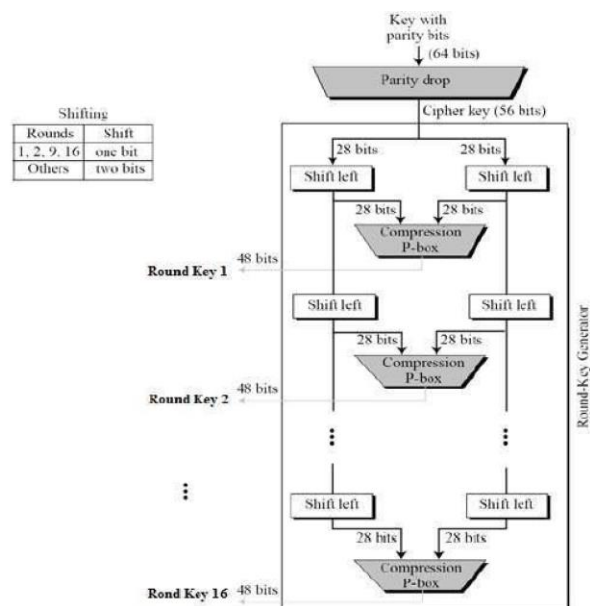
There are total of eight S-box tables. The output of all eight s-boxes is then combined in to 32-bit section.

Straight Permutation – The 32-bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

| | | | | | | | |
|----|----|----|----|----|----|----|----|
| 16 | 07 | 20 | 21 | 29 | 12 | 28 | 17 |
| 01 | 15 | 23 | 26 | 05 | 18 | 31 | 10 |
| 02 | 08 | 24 | 14 | 32 | 27 | 03 | 09 |
| 19 | 13 | 30 | 06 | 22 | 11 | 04 | 25 |

Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration –



Source Code:

```
#include <stdio.h>
#include <fstream>
#include <string.h>
#include <iostream>
#include <stdlib.h>
using namespace std;

int key[64] =
{
    0, 0, 0, 1, 0, 0, 1, 1,
    0, 0, 1, 1, 0, 1, 0, 0,
```

```

0, 1, 0, 1, 0, 1, 1, 1,
0, 1, 1, 1, 1, 0, 0, 1,
1, 0, 0, 1, 1, 0, 1, 1,
1, 0, 1, 1, 1, 1, 0, 0,
1, 1, 0, 1, 1, 1, 1, 1,
1, 1, 1, 1, 0, 0, 0, 1};

```

```

class Des
{
public:
    int keyi[16][48], total[64], left[32], right[32], ck[28], dk[28],
    expansion[48], z[48], xor1[48], sub[32], p[32], xor2[32], temp[64], pc1[56],
    ip[64], inv[8][8];

    char final[1000];
    void IP();
    void PermChoice1();
    void PermChoice2();
    void Expansion();
    void inverse();
    void xor_two();
    void xor_oneE(int);
    void xor_oneD(int);
    void substitution();
    void permutation();
    void keygen();
    char *Encrypt(char *);
    char *Decrypt(char *);
};

void Des::IP() // Initial Permutation
{
    int k = 58, i;
    for (i = 0; i < 32; i++)
    {
        ip[i] = total[k - 1];
        if (k - 8 > 0)
            k = k - 8;
        else
            k = k + 58;
    }
    k = 57;
    for (i = 32; i < 64; i++)
    {
        ip[i] = total[k - 1];
        if (k - 8 > 0)

```

```

        k = k - 8;
    else
        k = k + 58;
    }
}
void Des::PermChoice1() // Permutation Choice-1
{
    int k = 57, i;
    for (i = 0; i < 28; i++)
    {
        pc1[i] = key[k - 1];
        if (k - 8 > 0)
            k = k - 8;
        else
            k = k + 57;
    }
    k = 63;
    for (i = 28; i < 52; i++)
    {
        pc1[i] = key[k - 1];
        if (k - 8 > 0)
            k = k - 8;
        else
            k = k + 55;
    }
    k = 28;
    for (i = 52; i < 56; i++)
    {
        pc1[i] = key[k - 1];
        k = k - 8;
    }
}
void Des::Expansion() // Expansion Function applied on `right' half
{
    int exp[8][6], i, j, k;
    for (i = 0; i < 8; i++)
    {
        for (j = 0; j < 6; j++)
        {
            if ((j != 0) || (j != 5))
            {
                k = 4 * i + j;
                exp[i][j] = right[k - 1];
            }
            if (j == 0)

```

```

        {
            k = 4 * i;
            exp[i][j] = right[k - 1];
        }
        if (j == 5)
        {
            k = 4 * i + j;
            exp[i][j] = right[k - 1];
        }
    }
}
exp[0][0] = right[31];
exp[7][5] = right[0];
k = 0;
for (i = 0; i < 8; i++)
    for (j = 0; j < 6; j++)
        expansion[k++] = exp[i][j];
}
void Des::PermChoice2()
{
    int per[56], i, k;
    for (i = 0; i < 28; i++)
        per[i] = ck[i];
    for (k = 0, i = 28; i < 56; i++)
        per[i] = dk[k++];
    z[0] = per[13];
    z[1] = per[16];
    z[2] = per[10];
    z[3] = per[23];
    z[4] = per[0];
    z[5] = per[4];
    z[6] = per[2];
    z[7] = per[27];
    z[8] = per[14];
    z[9] = per[5];
    z[10] = per[20];
    z[11] = per[9];
    z[12] = per[22];
    z[13] = per[18];
    z[14] = per[11];
    z[15] = per[3];
    z[16] = per[25];
    z[17] = per[7];
    z[18] = per[15];
    z[19] = per[6];

```

```

    z[20] = per[26];
    z[21] = per[19];
    z[22] = per[12];
    z[23] = per[1];
    z[24] = per[40];
    z[25] = per[51];
    z[26] = per[30];
    z[27] = per[36];
    z[28] = per[46];
    z[29] = per[54];
    z[30] = per[29];
    z[31] = per[39];
    z[32] = per[50];
    z[33] = per[44];
    z[34] = per[32];
    z[35] = per[47];
    z[36] = per[43];
    z[37] = per[48];
    z[38] = per[38];
    z[39] = per[55];
    z[40] = per[33];
    z[41] = per[52];
    z[42] = per[45];
    z[43] = per[41];
    z[44] = per[49];
    z[45] = per[35];
    z[46] = per[28];
    z[47] = per[31];
}
void Des::xor_oneE(int round) // for Encrypt
{
    int i;
    for (i = 0; i < 48; i++)
        xor1[i] = expansion[i] ^ keyi[round - 1][i];
}
void Des::xor_oneD(int round) // for Decrypt
{
    int i;
    for (i = 0; i < 48; i++)
        xor1[i] = expansion[i] ^ keyi[16 - round][i];
}
void Des::substitution()
{
    int s1[4][16] =
        {

```

```

        14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7,
        0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8,
        4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0,
        15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13};
int s2[4][16] =
{
    15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10,
    3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5,
    0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15,
    13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9};
int s3[4][16] =
{
    10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8,
    13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1,
    13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7,
    1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12};
int s4[4][16] =
{
    7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15,
    13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9,
    10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4,
    3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14};
int s5[4][16] =
{
    2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9,
    14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6,
    4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14,
    11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3};
int s6[4][16] =
{
    12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11,
    10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8,
    9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6,
    4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13};
int s7[4][16] =
{
    4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1,
    13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6,
    1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2,
    6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12};
int s8[4][16] =
{
    13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7,
    1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2,
    7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8,

```



```

    2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11};
int a[8][6], k = 0, i, j, p, q, count = 0, g = 0, v;
for (i = 0; i < 8; i++)
{
    for (j = 0; j < 6; j++)
    {
        a[i][j] = xor1[k++];
    }
}
for (i = 0; i < 8; i++)
{
    p = 1;
    q = 0;
    k = (a[i][0] * 2) + (a[i][5] * 1);
    j = 4;
    while (j > 0)
    {
        q = q + (a[i][j] * p);
        p = p * 2;
        j--;
    }
    count = i + 1;
    switch (count)
    {
        case 1:
            v = s1[k][q];
            break;
        case 2:
            v = s2[k][q];
            break;
        case 3:
            v = s3[k][q];
            break;
        case 4:
            v = s4[k][q];
            break;
        case 5:
            v = s5[k][q];
            break;
        case 6:
            v = s6[k][q];
            break;
        case 7:
            v = s7[k][q];
            break;
    }
}

```

```

        case 8:
            v = s8[k][q];
            break;
        }
        int d, i = 3, a[4];
        while (v > 0)
        {
            d = v % 2;
            a[i--] = d;
            v = v / 2;
        }
        while (i >= 0)
        {
            a[i--] = 0;
        }
        for (i = 0; i < 4; i++)
            sub[g++] = a[i];
    }
}

void Des::permutation()
{
    p[0] = sub[15];
    p[1] = sub[6];
    p[2] = sub[19];
    p[3] = sub[20];
    p[4] = sub[28];
    p[5] = sub[11];
    p[6] = sub[27];
    p[7] = sub[16];
    p[8] = sub[0];
    p[9] = sub[14];
    p[10] = sub[22];
    p[11] = sub[25];
    p[12] = sub[4];
    p[13] = sub[17];
    p[14] = sub[30];
    p[15] = sub[9];
    p[16] = sub[1];
    p[17] = sub[7];
    p[18] = sub[23];
    p[19] = sub[13];
    p[20] = sub[31];
    p[21] = sub[26];
    p[22] = sub[2];
    p[23] = sub[8];
}

```

```

    p[24] = sub[18];
    p[25] = sub[12];
    p[26] = sub[29];
    p[27] = sub[5];
    p[28] = sub[21];
    p[29] = sub[10];
    p[30] = sub[3];
    p[31] = sub[24];
}
void Des::xor_two()
{
    int i;
    for (i = 0; i < 32; i++)
    {
        xor2[i] = left[i] ^ p[i];
    }
}
void Des::inverse()
{
    int p = 40, q = 8, k1, k2, i, j;
    for (i = 0; i < 8; i++)
    {
        k1 = p;
        k2 = q;
        for (j = 0; j < 8; j++)
        {
            if (j % 2 == 0)
            {
                inv[i][j] = temp[k1 - 1];
                k1 = k1 + 8;
            }
            else if (j % 2 != 0)
            {
                inv[i][j] = temp[k2 - 1];
                k2 = k2 + 8;
            }
        }
        p = p - 1;
        q = q - 1;
    }
}
char *Des::Encrypt(char *Text1)
{
    int i, a1, j, nB, m, iB, k, K, B[8], n, t, d, round;
    char *Text = new char[1000];

```

```

strcpy(Text, Text1);
i = strlen(Text);
int mc = 0;
a1 = i % 8;
if (a1 != 0)
    for (j = 0; j < 8 - a1; j++, i++)
        Text[i] = ' ';
Text[i] = '\\0';
keygen();
for (iB = 0, nB = 0, m = 0; m < (strlen(Text) / 8); m++) // Repeat for
TextLenth/8 times.
{
    for (iB = 0, i = 0; i < 8; i++, nB++)
    {
        n = (int)Text[nB];
        for (K = 7; n >= 1; K--)
        {
            B[K] = n % 2; // Converting 8-Bytes to 64-bit Binary Format
            n /= 2;
        }
        for (; K >= 0; K--)
            B[K] = 0;
        for (K = 0; K < 8; K++, iB++)
            total[iB] = B[K]; // Now `total' contains the 64-Bit binary
format of 8-Bytes
    }
    IP(); // Performing initial permutation on `total[64]'
    for (i = 0; i < 64; i++)
        total[i] = ip[i]; // Store values of ip[64] into total[64]
    for (i = 0; i < 32; i++)
        left[i] = total[i]; // +--> left[32]
    // total[64]--|
    for (; i < 64; i++)
        right[i - 32] = total[i]; // +--> right[32]
    for (round = 1; round <= 16; round++)
    {
        Expansion(); // Performing expansion on `right[32]' to get
`expansion[48]'
        xor_oneE(round); // Performing XOR operation on expansion[48],z[48]
to get xor1[48]
        substitution(); // Perform substitution on xor1[48] to get sub[32]
        permutation(); // Performing Permutation on sub[32] to get p[32]
        xor_two(); // Performing XOR operation on left[32],p[32] to get
xor2[32]
        for (i = 0; i < 32; i++)

```

```

        left[i] = right[i]; // Dumping right[32] into left[32]
    for (i = 0; i < 32; i++)
        right[i] = xor2[i]; // Dumping xor2[32] into right[32]
    }
    for (i = 0; i < 32; i++)
        temp[i] = right[i]; // Dumping -->[ swap32bit ]
    for (; i < 64; i++)
        temp[i] = left[i - 32]; // left[32],right[32] into temp[64]
    inverse(); // Inversing the bits of temp[64] to get
inv[8][8]
/* Obtaining the Cypher-Text into final[1000]*/
k = 128;
d = 0;
for (i = 0; i < 8; i++)
{
    for (j = 0; j < 8; j++)
    {
        d = d + inv[i][j] * k;
        k = k / 2;
    }
    final[mc++] = (char)d;
    k = 128;
    d = 0;
}
} // for loop ends here
final[mc] = '\0';
return (final);
}
char *Des::Decrypt(char *Text1)
{
    int i, a1, j, nB, m, iB, k, K, B[8], n, t, d, round;
    char *Text = new char[1000];
    unsigned char ch;
    strcpy(Text, Text1);
    i = strlen(Text);
    keygen();
    int mc = 0;
    for (iB = 0, nB = 0, m = 0; m < (strlen(Text) / 8); m++) // Repeat for
TextLenth/8 times.
    {
        for (iB = 0, i = 0; i < 8; i++, nB++)
        {
            ch = Text[nB];
            n = (int)ch; //(int)Text[nB];
            for (K = 7; n >= 1; K--)

```

```

    {
        B[K] = n % 2; // Converting 8-Bytes to 64-bit Binary Format
        n /= 2;
    }
    for (; K >= 0; K--)
        B[K] = 0;
    for (K = 0; K < 8; K++, iB++)
        total[iB] = B[K]; // Now `total' contains the 64-Bit binary
format of 8-Bytes
    }
    IP(); // Performing initial permutation on `total[64]'
    for (i = 0; i < 64; i++)
        total[i] = ip[i]; // Store values of ip[64] into total[64]
    for (i = 0; i < 32; i++)
        left[i] = total[i]; // +--> left[32]
    // total[64]--|
    for (; i < 64; i++)
        right[i - 32] = total[i]; // +--> right[32]
    for (round = 1; round <= 16; round++)
    {
        Expansion(); // Performing expansion on `right[32]' to get
`expansion[48]'
        xor_oneD(round);
        substitution(); // Perform substitution on xor1[48] to get sub[32]
        permutation(); // Performing Permutation on sub[32] to get p[32]
        xor_two();      // Performing XOR operation on left[32],p[32] to get
xor2[32]
        for (i = 0; i < 32; i++)
            left[i] = right[i]; // Dumping right[32] into left[32]
        for (i = 0; i < 32; i++)
            right[i] = xor2[i]; // Dumping xor2[32] into right[32]
    }
    // rounds end here
    for (i = 0; i < 32; i++)
        temp[i] = right[i]; // Dumping -->[ swap32bit ]
    for (; i < 64; i++)
        temp[i] = left[i - 32]; // left[32],right[32] into temp[64]
    inverse(); // Inversing the bits of temp[64] to get
inv[8][8]
    /* Obtaining the Cypher-Text into final[1000]*/
    k = 128;
    d = 0;
    for (i = 0; i < 8; i++)
    {
        for (j = 0; j < 8; j++)
        {

```

```

        d = d + inv[i][j] * k;
        k = k / 2;
    }
    final[mc++] = (char)d;
    k = 128;
    d = 0;
}
} // for loop ends here
final[mc] = '\0';
char *final1 = new char[1000];
for (i = 0, j = strlen(Text); i < strlen(Text); i++, j++)
    final1[i] = final[j];
final1[i] = '\0';
return (final);
}
int main()
{
    Des d1, d2;
    char *str = new char[1000];
    char *str1 = new char[1000];
    // strcpy(str,"PHOENIX it &ece solutions.");
    cout << "Enter a string : ";
    cin >> str;
    str1 = d1.Encrypt(str);
    cout << "\n/p Text: " << str << endl;
    cout << "\nCypher : " << str1 << endl;
    // ofstreamfout("out2_fil.txt"); fout<<str1; fout.close();
    cout << "\n/o/p Text: " << d2.Decrypt(str1) << endl;
}
void Des::keygen()
{
    PermChoice1();
    int i, j, k = 0;
    for (i = 0; i < 28; i++)
    {
        ck[i] = pc1[i];
    }
    for (i = 28; i < 56; i++)
    {
        dk[k] = pc1[i];
        k++;
    }
    int noshift = 0, round;
    for (round = 1; round <= 16; round++)
    {

```

```

    if (round == 1 || round == 2 || round == 9 || round == 16)
        noshift = 1;
    else
        noshift = 2;
    while (noshift > 0)
    {
        int t;
        t = ck[0];
        for (i = 0; i < 28; i++)
            ck[i] = ck[i + 1];
        ck[27] = t;
        t = dk[0];
        for (i = 0; i < 28; i++)
            dk[i] = dk[i + 1];
        dk[27] = t;
        noshift--;
    }
    PermChoice2();
    for (i = 0; i < 48; i++)
        keyi[round - 1][i] = z[i];
}
}

```

Output:

```

PS E:\sem 7\IS\codes> & .\"des.exe"
Enter a string : Syeda Reeha Quasar 14114802719

i/p Text: Syeda
Cypher : 4≤||≈ofY

o/p Text: Syeda
PS E:\sem 7\IS\codes> & .\"des.exe"
Enter a string : SyedaReehaQuasar

i/p Text: SyedaReehaQuasar
Cypher : ñ«■:ß!{NQÜÄâR||δ

o/p Text: SyedaReehaQuasar
PS E:\sem 7\IS\codes> & .\"des.exe"
Enter a string : reeha

i/p Text: reeha
Cypher : 8||±5▲W

o/p Text: reeha
PS E:\sem 7\IS\codes> 

```


EXPERIMENT – 6

Aim:

To implement Diffie–Hellman algorithm.

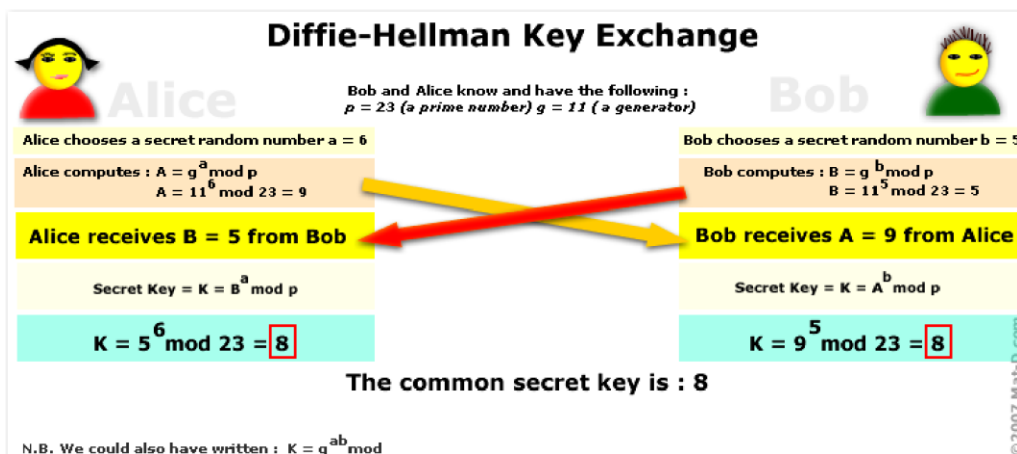
Theory:

Diffie-Hellman is a way of generating a shared secret between two people in such a way that the secret can't be seen by observing the communication. That's an important distinction: You're not sharing information during the key exchange, you're creating a key together.

This is particularly useful because you can use this technique to create an encryption key with someone, and then start encrypting your traffic with that key. And even if the traffic is recorded and later analysed, there's absolutely no way to figure out what the key was, even though the exchanges that created it may have been visible. This is where perfect forward secrecy comes from. Nobody analysing the traffic at a later date can break in because the key was never saved, never transmitted, and never made visible anywhere.

The Diffie-Hellman algorithm is being used to establish a shared secret that can be used for secret communications while exchanging data over a public network using the elliptic curve to generate points and get the secret key using the parameters.

- For the sake of simplicity and practical implementation of the algorithm, we will consider only 4 variables one prime P and G (a primitive root of P) and two private values a and b .
- P and G are both publicly available numbers. Users (say Alice and Bob) pick private values a and b and they generate a key and exchange it publicly, the opposite person received the key and from that generates a secret key after which they have the same secret key to encrypt.



Source Code:

```
#include <stdio.h>
#include <math.h>

// Power function to return value of  $a^b \bmod P$ 
long long int power(long long int a, long long int b,
                    long long int P)
{
    if (b == 1)
        return a;
    else
        return (((long long int)pow(a, b)) % P);
}

// Driver program
int main()
{
    long long int P, G, x, a, y, b, ka, kb;
    // Both the persons will be agreed upon the
    // public keys G and P
    P = 23; // A prime number P is taken
    printf("The value of P : %lld\n", P);
    G = 9; // A primitive root for P, G is taken
    printf("The value of G : %lld\n\n", G);
    // Alice will choose the private key a
    a = 4; // a is the chosen private key
    printf("The private key a for Alice : %lld\n", a);
    x = power(G, a, P); // gets the generated key
    // Bob will choose the private key b
    b = 3; // b is the chosen private key
    printf("The private key b for Bob : %lld\n\n", b);
    y = power(G, b, P); // gets the generated key
    // Generating the secret key after the exchange
    // of keys
    ka = power(y, a, P); // Secret key for Alice
    kb = power(x, b, P); // Secret key for Bob
    printf("Secret key for the Alice is : %lld\n", ka);
    printf("Secret Key for the Bob is : %lld\n", kb);

    printf("\n Syeda Reeha Quasar - 14114802719 \n");
    return 0;
}
```

Output:

```
PS E:\sem 7\IS\codes> & .\"dh.exe"
The value of P : 23
The value of G : 9

The private key a for Alice : 4
The private key b for Bob : 3

Secret key for the Alice is : 9
Secret Key for the Bob is : 9

Syeda Reeha Quasar - 14114802719
PS E:\sem 7\IS\codes> █
```

Result: Diffie-Hellman technique was implemented and understood.

EXPERIMENT – 7

Aim:

Make a study of anyone simulation tool based on parameters of information security.

Theory:

NeSSi consists of three distinct components, the Graphical User Interface, the simulation backend and the result database. Each of these modules may be run on separate machines depending on the computational requirements; furthermore, this modular design facilitates network security researchers using NeSSi to easily exchange.

Graphical User Interface

The graphical frontend of NeSSi allows the user to create and edit network topologies, attach runtime information, and schedule them for execution at the simulation backend. On the other hand, finished (or even currently executing, long-running) simulations can be retrieved from the database server and the corresponding simulation results are visualized in the GUI.

Simulation Backend

The actual simulation is performed on machine with hardware dedicated solely to this purpose, the simulation backend. At the DAI-Labor for example, the NeSSi simulation backend runs on a Sun XFire 4600 blade server (8 blades, 8 cores per blade). Once a session is submitted for execution from the GUI, the simulation backend parses the desired session parameters (which event types to log, how many runs to execute etc.), creates a corresponding simulation environment, sets up the database connection and schedules the simulation to run as soon as the necessary processing resources are available.

Parallel Execution Model. Simulations in large-scale networks are very costly in terms of processing time and memory consumption. Therefore, NeSSi has been designed as a distributed simulation, allowing the subdivision of tasks to different computers and processes in a parallel-execution model.

Discrete Event Simulation. NeSSi² is a discrete-event-based simulation tool, which allows to plan and to schedule time-based events such as network failures, attacks, etc.

Simulation Result Database Server

In NeSSi, we refer to a scenario where we generate traffic via pre-defined profiles on a single network over a certain amount of time as a *session*. The accurate reproduction of a session enables users to use different detection methods or various deployments of detection units for the same traffic data set. This allows the comparison of performance

and detection efficiency of different security framework setups.

For these purposes, we use a **distributed database** in which the traffic generated during a session is stored. For each session, the agents log the traffic and detection data and send it to the database that occurs in a simulated scenario between a start and end time. The data types to be logged are specified by the user in the session parameters. The network model is saved in an XML file. This network file is stored and annotated with a version number based on its hash code in order to link a network uniquely to a session. Additionally, attack related events can be stored in the database for evaluation purposes.

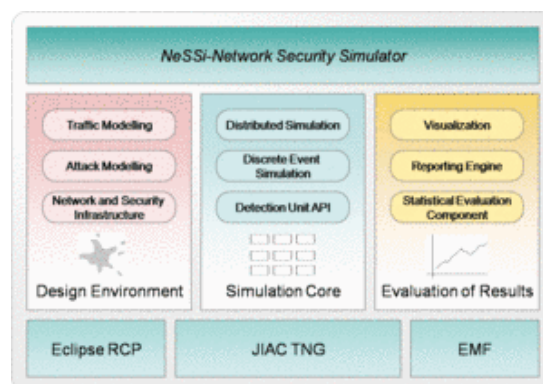
Standard Components and Plugin API

NeSSi² has been designed as a modularized application. Building on the Eclipse framework, it uses the inherent plugin mechanism to allow users to easily extend the functionality of NeSSi² and share it with other developers.

Often, security researchers have very specific demands regarding the protocols and features the simulation tool needs to offer. Naturally, NeSSi² provides a rich set of basic protocols and detection unit implementations; nevertheless, the special needs of various application areas (wireless networks, sensor networks, MPLS etc.) necessitates a plugin API to allow the user to adapt NeSSi² to his needs and add the functionality that is not provided by NeSSi out-of-the-box.

Hence, the NeSSi² extension API allows the creation of

- New device types with user-defined properties
- New protocols defining the behavior of the network at runtime
- Application definitions, allowing dynamic behavior to be defined, attached to a device or link, and scheduled for execution in the simulation



EXPERIMENT – 8

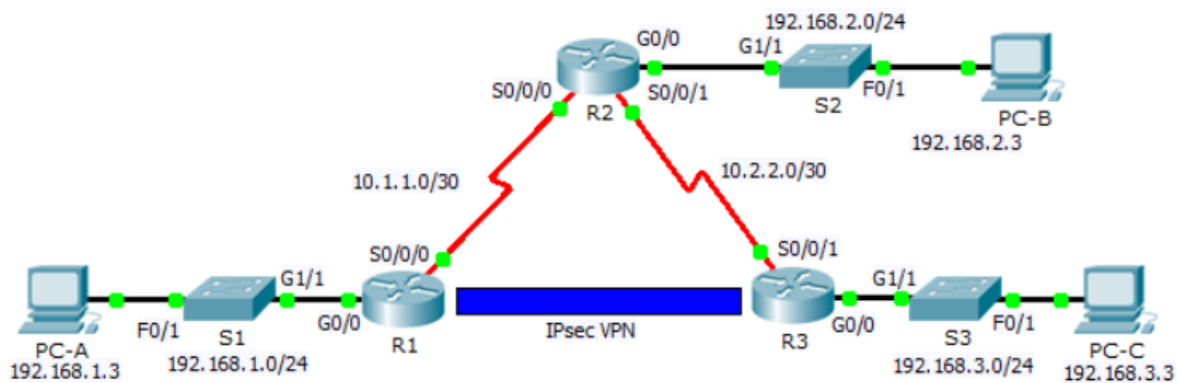
Aim:

Implement VPN through Packet-Tracer or any other network simulator tool.

Theory:

SoftEther VPN can construct distributed virtual Ethernet segment. If you can make some geologically distributed computers enable to communicate each other as if they are connected to the single Ethernet network, using SoftEther VPN is the easiest way.

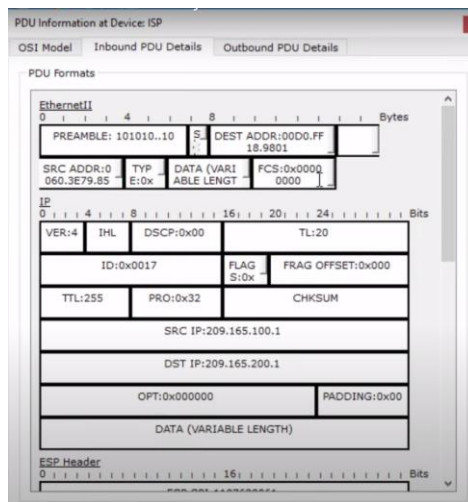
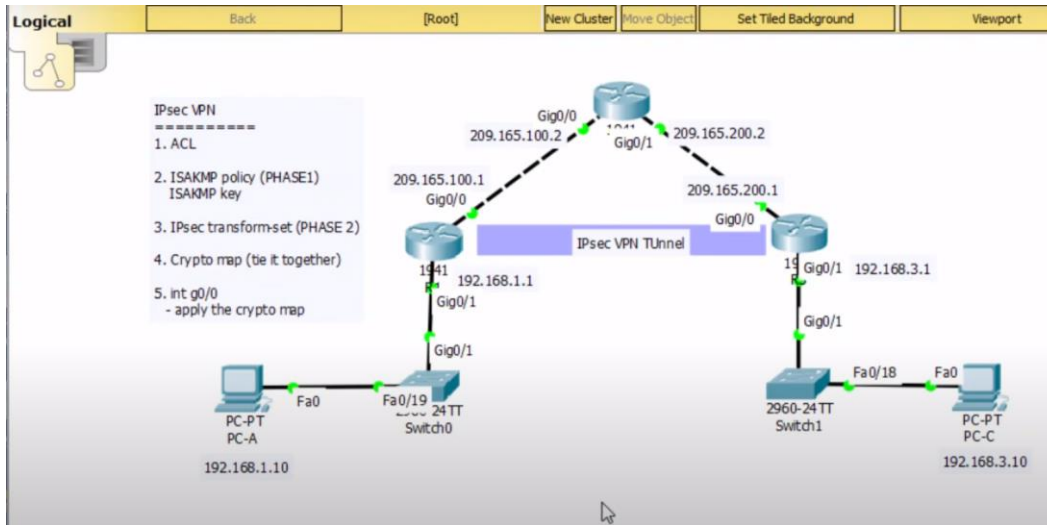
Topology



Addressing Table

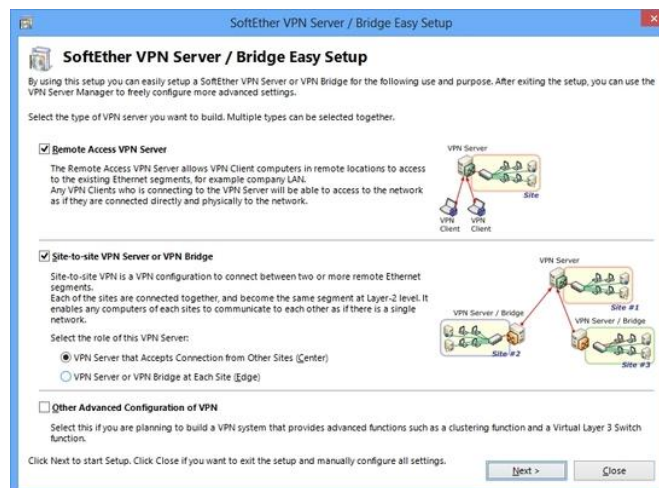
| Device | Interface | IP Address | Subnet Mask | Default Gateway |
|--------|-----------|-------------|-----------------|-----------------|
| R1 | G0/0 | 192.168.1.1 | 255.255.255.0 | N/A |
| | S0/0/0 | 10.1.1.2 | 255.255.255.252 | N/A |
| R2 | G0/0 | 192.168.2.1 | 255.255.255.0 | N/A |
| | S0/0/0 | 10.1.1.1 | 255.255.255.252 | N/A |
| | S0/0/1 | 10.2.2.1 | 255.255.255.252 | N/A |
| R3 | G0/0 | 192.168.3.1 | 255.255.255.0 | N/A |
| | S0/0/1 | 10.2.2.2 | 255.255.255.252 | N/A |
| PC-A | NIC | 192.168.1.3 | 255.255.255.0 | 192.168.1.1 |
| PC-B | NIC | 192.168.2.3 | 255.255.255.0 | 192.168.2.1 |
| PC-C | NIC | 192.168.3.3 | 255.255.255.0 | 192.168.3.1 |

First, set up a VPN Server. Next, set up VPN Clients on each member PCs. Finally start VPN connections on each VPN client. Then each clients can use any kinds of IP-based or Ethernet-based protocols via the VPN even if they are distributed around the world.



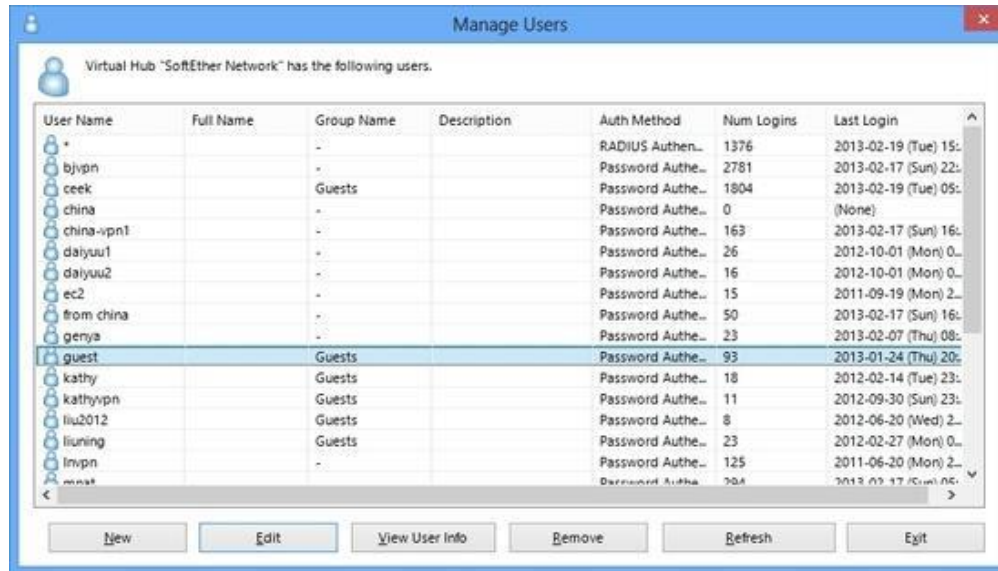
Step 1. Set up SoftEther VPN Server

Designate a computer in the group as the VPN Server. [Set up SoftEther VPN Server](#) on that computer. It is very easy by using Installer and Initial Setup Wizard based GUI.



Step 2. Create Users

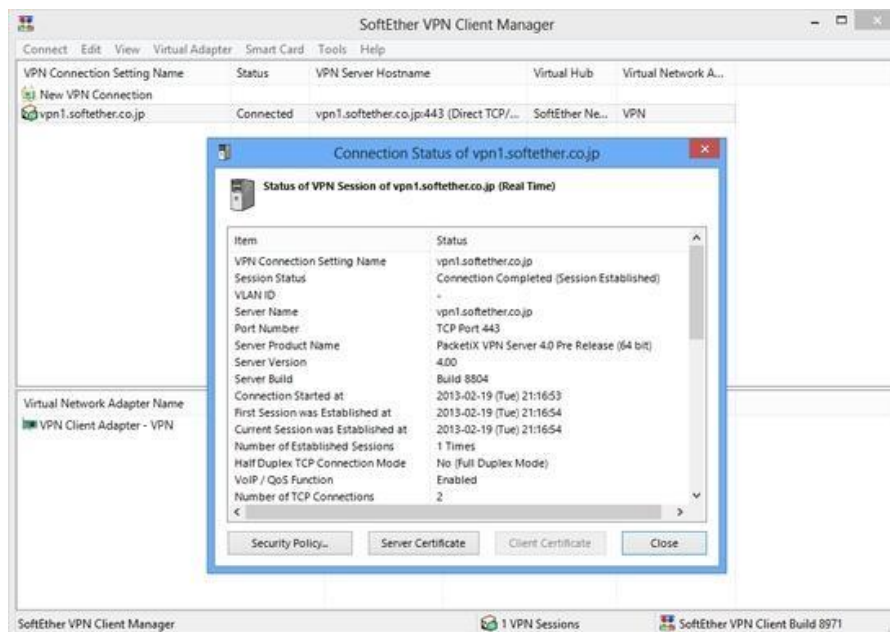
On the VPN Server you can add several user objects on the Virtual Hub. Each user object has a password. After that, distribute pairs of username and password to each member of the VPN.



Step 3. Set up VPN Client on Each Member's PC

On each member's PC install SoftEther VPN Client. Enter the server address, username and password for each PC.

If a member of the VPN is Mac OS X, iPhone or Android, set up L2TP/IPsec VPN client on each PC instead of SoftEther VPN. Another solution is to use OpenVPN Client on Mac OS X, iPhone or Android to connect to SoftEther VPN Server.

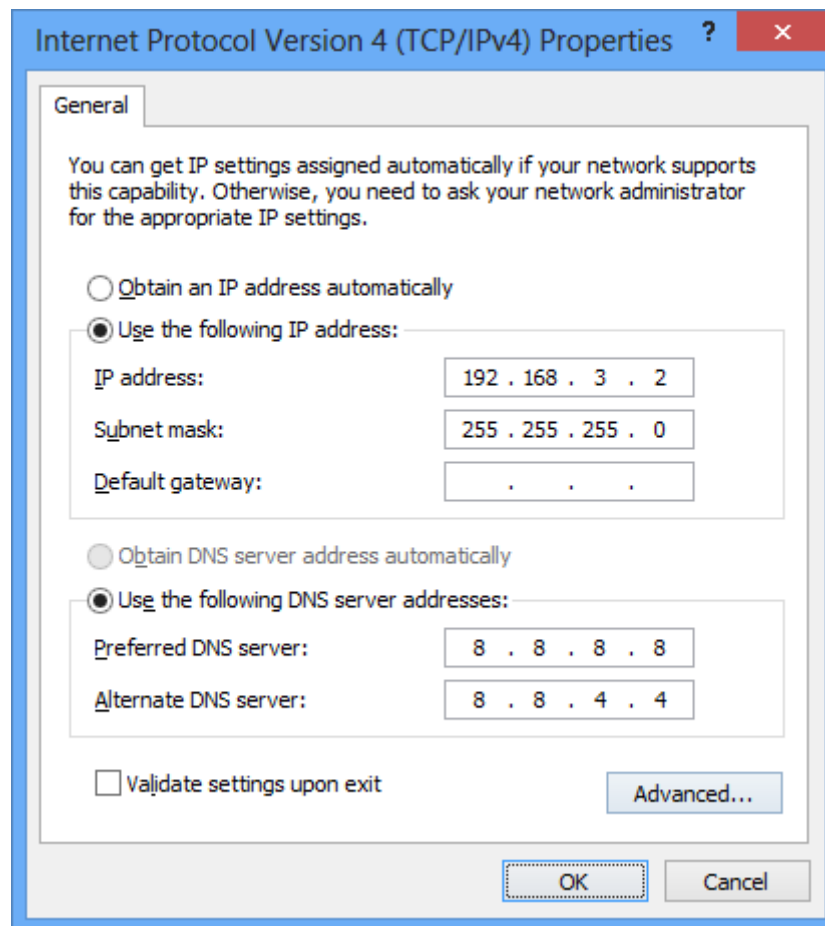


Step 4. Set up IP Addresses

The characteristics of SoftEther's virtual private network is exactly same to a physical Ethernet segment. So you should decide the IP addresses of every member PCs.

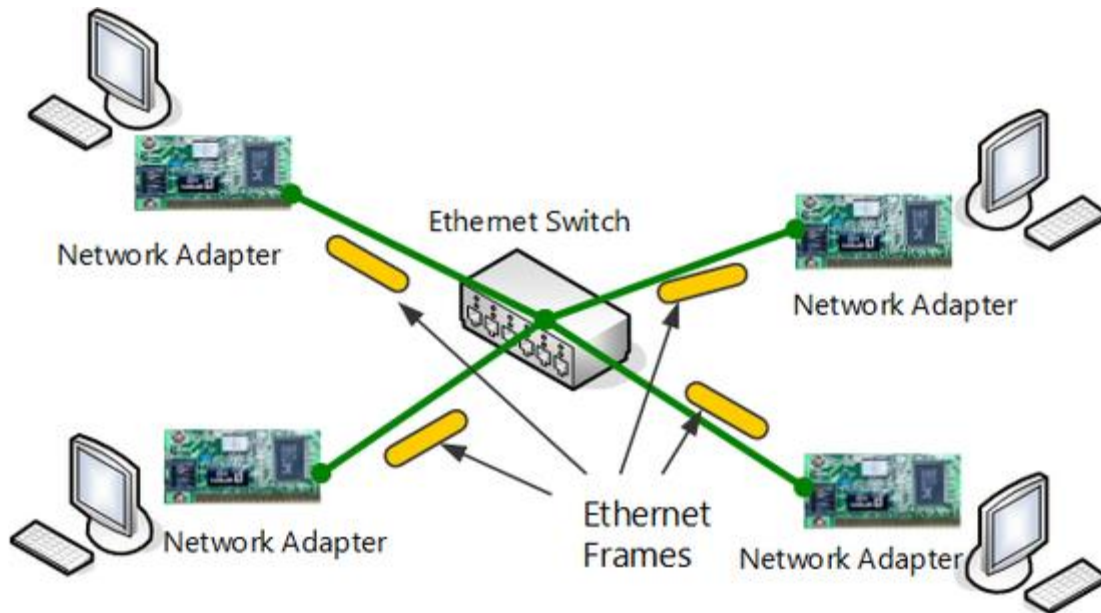
Like the physical Ethernet, the simplest way is to set up private IP addresses to each PC, for example 192.168.0.0/24. Make sure not to overlap to physical-using private IPs.

Another solution is to use DHCP server for automated IP address allocation. You can activate Virtual DHCP Server Function on the SoftEther VPN Server and it will distribute 192.168.30.0/24 by default.



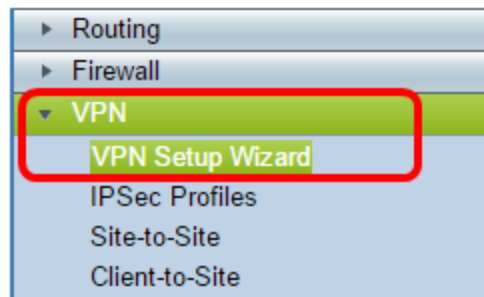
Step 5. Communicate Like Physical Ethernet

Once every computers are connected to the Virtual Hub on SoftEther VPN Server, all computers and smart-phones can now communicate mutually as if they are all connected to the single Ethernet network. You can enjoy File Sharing protocols, Remote Printing applications, Remote Desktop applications, SQL Database applications and any other LAN-based applications despite the distances and differences of physical location.



Configure VPN Connection using the Setup Wizard

Step 1. Log in to the router web-based utility and choose **Configuration Wizard**. Then click **Launch Wizard** under *VPN Setup Wizard* section.



Step 2. In the field provided, enter a name to identify this connection.

Give this connection a name:

Note: In this example, TestVPN is used.

Step 3. In the Interface area, click the drop-down menu and choose which interface you want to enable this connection. The options are:

- WAN1
- WAN2
- USB1
- USB2

Give this connection a name: Laboratory

Interface: WAN1

Next Cancel

Note: In this example, WAN1 is used.

Step 4. Click **Next**.

Give this connection a name: Laboratory

Interface: WAN1

Next Cancel

Step 5. Choose the Remote Connection Type by clicking on the drop-down arrow. The options are:

- IP Address — Choose this option if you want to use the IP address of the remote router at the other end of the VPN tunnel.
- FQDN — (Fully Qualified Domain Name) Choose this option if you want to use the domain name of the remote router at the other end of the VPN tunnel.

Remote Connection Type: IP Address

Remote Connection: IP Address

Note: In this example, IP Address is chosen.

Step 6. Enter the WAN IP address of the Remote Connection in the field provided and then click **Next**.

Remote Connection Type: IP Address

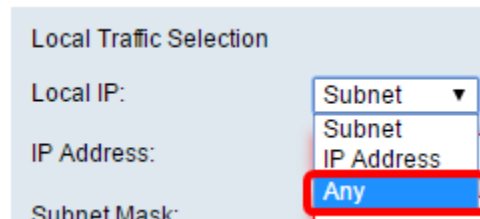
Remote Connection: 10.10.10.1

Back Next Cancel

Note: In this example, 128.123.2.1 is used.

Step 7. Under the Local Traffic Selection area, click on the drop-down to choose the Local IP. The options are:

- Subnet — Choose this if you want to enter both the IP address and Subnet mask of the Local network.
- IP Address — Choose this if you want to enter just the IP address of the local network.
- Any — Choose this if you want any of the two.



Local Traffic Selection

Local IP: Subnet ▼

IP Address: Subnet

Subnet Mask: IP Address

Any

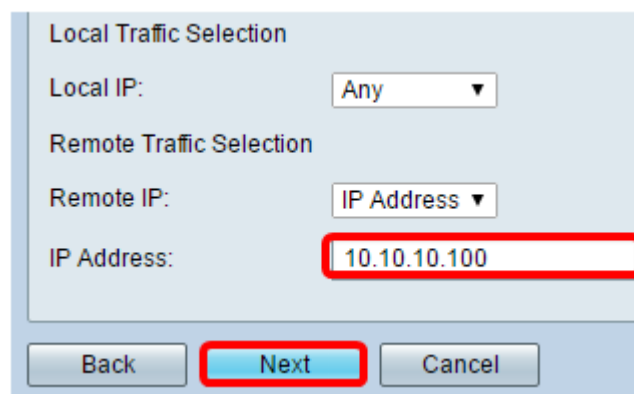
Note: In this example, Any is chosen.

Step 8. Under Remote Traffic Selection area, click the drop-down arrow to choose the Remote IP. Enter the remote IP address and subnet mask in the field provided then click **Next**. The options are:

- Subnet — Choose this if you want to enter both the IP address and Subnet mask of the remote network.
- IP Address — Choose this if you want to enter just the IP address of the remote network.

Note: In this example, Subnet is chosen. 10.10.10.0 was entered as the IP address and 255.255.255.0 was entered as the subnet mask.

Step 9. Click the drop-down arrow in the IPsec Profile area to choose which profile to use.



Local Traffic Selection

Local IP: Any ▼

Remote Traffic Selection

Remote IP: IP Address ▼

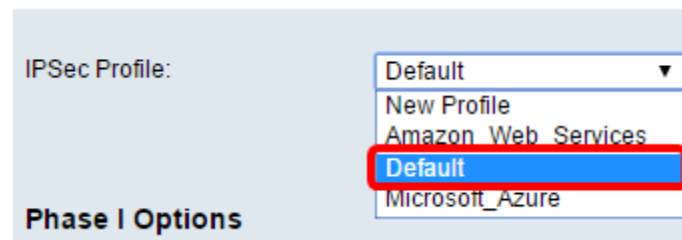
IP Address: 10.10.10.100

Back Next Cancel

Note: In this example, Default is chosen.

Step 10. Under the Phase 1 Options area, enter the pre-shared key for this connection in the field provided. This is the pre-shared key to be used to authenticate the remote Internet Key Exchange (IKE) peer. Both ends of the VPN tunnel must use the same pre-shared key. Up to 30 characters or hexadecimal values are allowed to be used for this key.

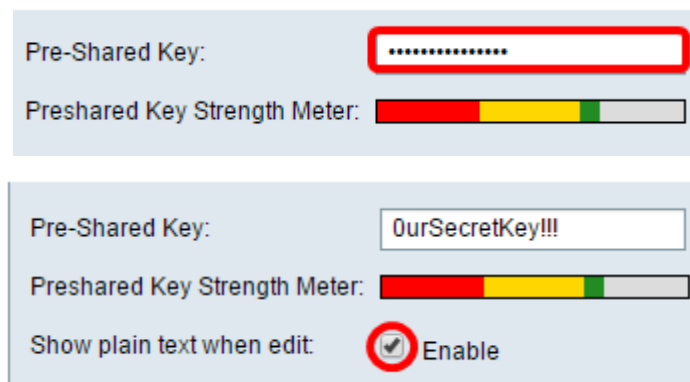
Note: It is highly advised to change the pre-shared key regularly to maintain the security of your VPN connection.



Note: The Preshared Key Strength Meter indicates the strength of the key you have entered based on the following:

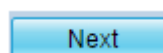
- Red — The password is weak.
- Amber — The password is fairly strong.
- Green — The password is strong.

Step 11. (Optional) You can also check the **Enable** check box in the Show plain text when edit to see the password in plain text.



Step 12. Click **Next**.

Step 13. The page will then show all the configuration details of your VPN connection. Click **Submit**.



You should now have successfully configured VPN connection on the RV34x Series Router using the Setup Wizard. To successfully connect a site-to-site VPN, you will need to configure the Setup Wizard on the remote router.

Viva Questions

1. Do WIFI Certificates Replace Wireless Security Protocols Like Wpa2?

Ans: As the successor to WPA, the WPA2 standard was ratified by IEEE in 2004 as 802.11i. Like its predecessor, WPA2 also offers enterprise and personal modes.

2. If My Wireless Network Doesn't Have A Lot of Traffic, Is It Okay to Use WEP Because the Ivs Required to Crack the WEP Key Won't Be Generated?

Ans: WEP encrypts all traffic using a 64 or 128 bit key in hexadecimal. This is a static key, which means all traffic, no matter the device, is encrypted using the single key.

3. What Is the Difference Between Active and Passive WLAN Detection?

Ans. A client can use two scanning methods: active and passive. During an active scan, the client radio transmits a probe request and listens for a probe response from an AP. With a passive scan, the client radio listens on each channel for beacons sent periodically by an AP.

4. How Many types of extensible Authentication Protocols (eaps) Are supported by Wpa/wpa2 And What Are They?

Ans: The current WPA/WPA2 certified EAP standards are:

- EAP-TLS (originally certified protocol)
- EAP-TTLS/MSCHAPv2
- PEAPv0/EAP-MSCHAPv2
- PEAPv1/EAP-GTC
- EAP-SIM

5. What is the primary difference Between 802.11g And 802.11a?

Ans: 802.11n, which once Apple confusingly dubbed "AirPort Extreme" -- the same name the company gave to the earlier 802.11g specification -- is a wireless networking standard first supported by the [original Apple TV](#). Starting on July 20, 2011, Apple quietly started referring to this standard just as 802.11n "Wi-Fi" for simplicity.

Viva Questions

1. What is a network firewall?

Ans: A firewall is a security system that is placed between a trusted and an untrusted network. It blocks or allows traffic that enters or leaves the network based on pre-configured policies or rules.

2. What is synchronization and why is it important?

Ans: Synchronization in java is the capability to control the access of multiple threads to any shared resource. In the Multithreading concept, multiple threads try to access the shared resources at a time to produce inconsistent results.

3. What are the critical resources in a firewall?

Ans: Firewalls defend your computer or network from outside cyber attackers by filtering out dangerous or superfluous network traffic. Firewalls can also prevent harmful malware from gaining access to a computer or network through the internet.

4. What are some common attacks, and how can I protect my system against them?

Ans: A cyber attack is a deliberate exploitation of your systems and/or network. Cyber attacks use malicious code to compromise your computer, logic or data and steal, leak or hold your data hostage. Cyber attack prevention is essential for every business and organisation.

Here are some examples of common cyber attacks and types of data breaches:

- Identity theft, fraud, extortion
- Malware, phishing, spamming, spoofing, spyware, trojans and viruses
- Stolen hardware, such as laptops or mobile devices
- Denial-of-service and distributed denial-of-service attacks
- Breach of access
- Password sniffing
- System infiltration
- Website defacement
- Private and public Web browser exploits

- Instant messaging abuse
- Intellectual property (IP) theft or unauthorized access

5. What is the difference between gateway and firewall?

Ans: A gateway is used to link two separate networks together, allowing users to communicate across several networks. In contrast, a firewall secures a network by deciding which data packets are allowed to pass through a network

Viva Questions

1. What Is SQL Injection?

Ans: SQL injection is a code injection technique that might destroy your database. SQL injection is one of the most common web hacking techniques.

2. What Is A 'Threat Model'? How Do You Go About Designing One?

Ans: Threat modeling is a structured process with these objectives: identify security requirements, pinpoint security threats and potential vulnerabilities, quantify threat and vulnerability criticality, and prioritize remediation methods. Threat modeling methods create these artifacts: An abstraction of the system.

3. Can You Name the Three Parts of A TCP Handshake?

Ans: The Three parts are:

Step 1 (SYN).

Step 2 (SYN + ACK)

Step 3 (ACK)

4. What Are the Phases Of Network Penetration?

Ans: There are five penetration testing phases: reconnaissance, scanning, vulnerability assessment, exploitation, and reporting.

5. What are nodes and links?

Ans: **Node:** Any communicating device in a network is called a Node. Node is the point of intersection in a network. It can send/receive data and information within a network. Examples of the node can be computers, laptops, printers, servers, modems, etc.

Link: A link or edge refers to the connectivity between two nodes in the network. It includes the type of connectivity (wired or wireless) between the nodes and protocols used for one node to be able to communicate with the other.

Viva Questions

1. How Fast Is RSA?

Ans. The proposed hardware implementation of RSA algorithm has a maximum frequency of operation of 545 MHz and 298 MHz for the bit sizes of 8 and 64 respectively. Entire groups of users can use the same public exponent, each with a different modulus.

2. What Would It Take to Break RSA?

Ans. It would take a classical computer around 300 trillion years to break a RSA-2048 bit encryption key. here signatures. The obvious way to do this attack is to factor the public modulus, n , into its two prime factors, p and q .

3. Are Strong Primes Necessary In RSA?

Ans. They show that cycling attacks are extremely unlikely to be effective, as long as the primes used are large.

4. How Large A Modulus (key) Should Be Used In RSA?

Ans. The minimum size for secure RSA keys on the token key data set (TKDS) is 1024 bits and the size must be a multiple of 256.

5. How Large Should The Primes Be?

Ans. The recommended RSA modulus size for most settings is 2048 bits to 4096 bits. Thus, the primes to be generated need to be 1024 bit to 2048 bit long.

Viva Questions

1. Give a one- or two-sentence definition of Symmetric key cryptography.

Ans. In cryptography, a symmetric key is one that is used both to encrypt and decrypt information. This means that to decrypt information, one must have the same key that was used to encrypt it.

2. Give a one- or two-sentence definition Asymmetric key cryptography.

Ans. Asymmetric cryptography, also known as public-key cryptography, is a process that uses a pair of related keys -- one public key and one private key -- to encrypt and decrypt a message and protect it from unauthorized access or use.

3. How can a message between 2 people be authenticated using symmetric encryption?

Ans. By using symmetric encryption algorithms, data is "scrambled" so that it can't be understood by anyone who does not possess the secret key to decrypt it. Once the intended recipient who possesses the key has the message, the algorithm reverses its action so that the message is returned to its original readable form.

4. What is DES Encryption?

Ans. DES has been found vulnerable to very powerful attacks and therefore, the popularity of DES has been found slightly on the decline. DES is a block cipher and encrypts data in blocks of size of 64 bits each, which means 64 bits of plain text go as the input to DES, which produces 64 bits of ciphertext. The same algorithm and key are used for encryption and decryption, with minor differences. The key length is 56 bits. The basic idea is shown in the figure:

5. Give an example of DES encryption Step by Step?

Ans. Encryption:

After initial permutation: 14A7D67818CA18AD

After splitting: L0=14A7D678 R0=18CA18AD

Round 1 18CA18AD 5A78E394 194CD072DE8C
Round 2 5A78E394 4A1210F6 4568581ABCCE
Round 3 4A1210F6 B8089591 06EDA4ACF5B5
Round 4 B8089591 236779C2 DA2D032B6EE3
Round 5 236779C2 A15A4B87 69A629FEC913
Round 6 A15A4B87 2E8F9C65 C1948E87475E
Round 7 2E8F9C65 A9FC20A3 708AD2DDB3C0
Round 8 A9FC20A3 308BEE97 34F822F0C66D
Round 9 308BEE97 10AF9D37 84BB4473DCCC
Round 10 10AF9D37 6CA6CB20 02765708B5BF
Round 11 6CA6CB20 FF3C485F 6D5560AF7CA5
Round 12 FF3C485F 22A5963B C2C1E96A4BF3
Round 13 22A5963B 387CCDAA 99C31397C91F
Round 14 387CCDAA BD2DD2AB 251B8BC717D0
Round 15 BD2DD2AB CF26B472 3330C5D9A36D
Round 16 19BA9212 CF26B472 181C5D75C66D

Cipher Text: C0B7A8D05F3A829C

Decryption

After initial permutation: 19BA9212CF26B472

After splitting: L0=19BA9212 R0=CF26B472

Round 1 CF26B472 BD2DD2AB 181C5D75C66D
Round 2 BD2DD2AB 387CCDAA 3330C5D9A36D
Round 3 387CCDAA 22A5963B 251B8BC717D0
Round 4 22A5963B FF3C485F 99C31397C91F
Round 5 FF3C485F 6CA6CB20 C2C1E96A4BF3

Round 6 6CA6CB20 10AF9D37 6D5560AF7CA5

Round 7 10AF9D37 308BEE97 02765708B5BF

Round 8 308BEE97 A9FC20A3 84BB4473DCCC

Round 9 A9FC20A3 2E8F9C65 34F822F0C66D

Round 10 2E8F9C65 A15A4B87 708AD2DDB3C0

Round 11 A15A4B87 236779C2 C1948E87475E

Round 12 236779C2 B8089591 69A629FEC913

Round 13 B8089591 4A1210F6 DA2D032B6EE3

Round 14 4A1210F6 5A78E394 06EDA4ACF5B5

Round 15 5A78E394 18CA18AD 4568581ABCCE

Round 16 14A7D678 18CA18AD 194CD072DE8C

Plain Text: 123456ABCD132536

Viva Questions

1. How can a message between 2 people be authenticated using asymmetric encryption?

Ans. Asymmetric cryptography is typically used to authenticate data using digital signatures. A digital signature is a mathematical technique used to validate the authenticity and integrity of a message, software or digital document. It is the digital equivalent of a handwritten signature or stamped seal.

2. Discuss alternative ways that the Diffie-Hellman exchange can be secured against a "man-in-the-middle" attack.

Ans. Different ways are:

- Strong WEP/WAP encryption on access points
- Strong router login credentials
- Virtual private network
- Forces https
- Public key pair based authentication

3. The exercise is to calculate what the key would be for Alice and Bob using these known values:

- Prime number selected is $p = 337$
- Random number selected is $g = 3$ **Generator
- Alice random private number selected is $x_a = 7$
- Bob's random private number selected is $x_b = 11$

What is the key that Alice and Bob will use?

Ans. Alice: 241, Bob: -1

4. What are the HTTP and the HTTPS protocol?

Ans. HTTP is the HyperText Transfer Protocol which defines the set of rules and standards on how the information can be transmitted on the World Wide Web (WWW). It helps the web browsers and web servers for communication. It is a 'stateless protocol' where each command is independent with respect to the previous command. HTTP is an application layer protocol built upon the TCP. It uses port 80 by default.

HTTPS is the HyperText Transfer Protocol Secure or Secure HTTP. It is an advanced and secured version of HTTP. On top of HTTP, SSL/TLS protocol is used to provide security. It enables secure transactions by encrypting the communication and also helps identify network servers securely. It uses port 443 by default.

5. What is the DNS?

Ans. DNS is the Domain Name System. It is considered as the devices/services directory of the Internet. It is a decentralized and hierarchical naming system for devices/services connected to the Internet. It translates the domain names to their corresponding IPs. For e.g. interviewbit.com to 172.217.166.36. It uses port 53 by default.

Viva Questions

1. What is Cryptography?

Ans. Cryptography is the practice and study of techniques for securing information and communication mainly to protect the data from third parties that the data is not intended for.

2. What is the difference between Symmetric and Asymmetric encryption?

Ans.

| Basis of Comparison | Symmetric Encryption | Asymmetric Encryption |
|---------------------|--|--|
| Encryption key | Same key for encryption & decryption | Different keys for encryption & decryption |
| Performance | Encryption is fast but more vulnerable | Encryption is slow due to high computation |
| Algorithms | DES, 3DES, AES and RC4 | Diffie-Hellman, RSA |
| Purpose | Used for bulk data transmission | Often used for securely exchanging secret keys |

3. What is the difference between IDS and IPS?

Ans. IDS is Intrusion Detection System and it only detects intrusions and the administrator has to take care of preventing the intrusion. Whereas, in IPS i.e., Intrusion Prevention System, the system detects the intrusion and also takes actions to prevent the intrusion.

4. How is Encryption different from Hashing?

Ans. Both Encryption and Hashing are used to convert readable data into an unreadable format. The difference is that the encrypted data can be converted back to original data by the process of decryption but the hashed data cannot be converted back to original data.

5. What is the difference between VA(Vulnerability Assessment) and PT(Penetration Testing)?

Ans.

Vulnerability Assessment is the process of finding flaws on the target. Here, the organization knows that their system/network has flaws or weaknesses and want to find these flaws and prioritize the flaws for fixing.

Penetration Testing is the process of finding vulnerabilities on the target. In this case, the organization would have set up all the security measures they could think of and would want to test if there is any other way that their system/network can be hacked.

Viva Questions

1. if you receive an error related to Protocol Error is occurring, what all troubleshooting measures can be taken up?

Ans:

1. Make Sure SSL Is Installed on Your Website
2. Force HTTPS Connection on Your Website
3. Clear Your Browser's Cache

2. There is a large number of broadcast packets constantly being sent over the network. What should I check?

Ans: Some methods for dealing with DHCP-related broadcast storms:

- Stagger the enablement of a large group of devices that would otherwise all request an address through DHCP at the same time.
- Check to see if you're using a DHCP relay between some or all of your VLANs. DHCP relay routes broadcast packets between broadcast domains.
- Set up a maintenance window in Auvik to silence broadcast-related alerts during known periods of high-volume DHCP requests.

3. Which protocol is suited for communication over insecure channel?

Ans: MANA I Protocol

4. How many firewalls should be there in the network?

Ans: If your network is entirely client-protecting, or is client-protecting with just a few incoming services, such as email, then one firewall (or a pair of firewalls configured as a high-availability pair) is probably all you need.

5. What is the FTP protocol?

Ans. FTP is a File Transfer Protocol. It is an application layer protocol used to transfer files and data reliably and efficiently between hosts. It can also be used to download files from remote servers to your computer. It uses port 27 by default.