6 Web Security

Module 6

Syllabus

At the end of this unit, you should be able to understand and comprehend the following syllabus topics:

- Web Security Considerations
  - SSL
  - HTTPS
  - SSH
  - Secure Electronic Transaction (SET)
- User Authentication and Session Management
  - Cookies
  - Session Hijacking and Management
- Privacy on Web
- Web Browser Attacks
  - Account Harvesting
  - Web Bugs
  - Clickjacking
  - Cross-Site Request Forgery (CSRF)
  - Phishing and Pharming Techniques
  - DNS Attacks
- Email Attacks
- Web Service Security
  - Web Server Security as per OWASP
  - Firewalls
  - Penetration Testing

6.1 Web Security Considerations

- World Wide Web or just web is a collection of web servers that run several websites that hold the desired information. The Internet as a whole is a collection of such servers and various communication devices and protocols. You mostly use browsers (such as Chrome, Firefox, Internet Explorer, Safari, etc.) or applications (for example, Mobile Apps) to browse the web and fetch the desired information or just complete a desired interaction such as making a purchase.
- Let me pause you here and ask a simple question. Don't you feel that your interaction with the Internet (which generally is an insecure and unsafe place) should be protected? For example, if you type your Facebook password, should it be available to everyone on the network? To make a purchase when you provide your bank account information, isn't that information very confidential and requires secure handling as you pass it through your device all the way to the website? Yes, I am sure you understand that your interaction with the web requires security.

– In this unit, you would learn about several threats to web and what are the possible ways of mitigating the risks arising from them. You will also learn about some of major security protocols and mechanisms that help us to keep the web a secure and a safe place for everyone.

6.2 Secure Socket Layer (SSL)

– The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) are the most widely use web security protocol. It is essentially a protocol that provides a secure channel between two machines operating over the Internet or an internal network.

✍ ***Definition :*** *SSL is a cryptographic protocol designed to protect communication between two entities.*

– SSL underwent several revisions and is now followed by a more secure protocol called TLS. Following is a quick version history of SSL/TLS.

Protocol	Published	Status
SSL 1.0	Unpublished	Unpublished
SSL 2.0	1995	Obsoleted in 2011
SSL 3.0	1996	Obsoleted in 2015
TLS 1.0	1999	To be obsolete in 2020
TLS 1.1	2006	To be obsolete in 2020
TLS 1.2	2008	Currently good
TLS 1.3	2018	Currently good

6.2.1 Goals of SSL

1. **Cryptographic Security :** Establish and provide a secure connection between two parties.
2. **Interoperability :** Two unrelated applications should be able to establish SSL connection.
3. **Extensibility :** Provides a framework for using various algorithms and methods without changing the protocol.
4. **Efficiency :** Performance enhancement mechanism to avoid overloading the system when protocol is in use.

6.2.2 Overview of SSL protocol

– SSL protocol works in layers. At each layer, messages may include fields for length, description, and content. SSL takes messages to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, and transmits the result. On the other side, received data is decrypted, verified, decompressed, and reassembled and then delivered to higher level clients.

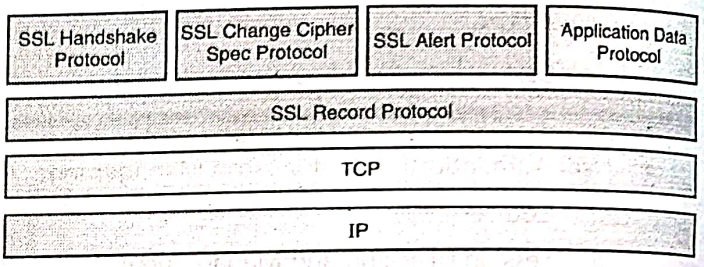


Fig. 6.2.1

Let's learn about each one of them in detail.

a. **Session and Connection States :** An SSL session is stateful which means that parameters negotiated during the session establishment persist (stay the same) until the session is terminated. The SSL handshake protocol coordinates the states of the client and server. It is thus important to preserve Session and Connection States.

The following table summarizes a Session State.

Fields	Purpose
Session Identifier	A session ID chosen by the server to identify an active or resumable session state
Peer Certificate	X.509 Certificate of the other party in the communication
Compression method	The algorithm used to compress data prior to encryption
Cipher Specification	Chosen encryption algorithm such as AES and a hash algorithm such as SHA
Master Secret	48-byte secret shared between the client and server
Is resumable	A Boolean flag indicating whether a session ID can be used to initiate new connections

The following table summarises a Connection State.

Fields	Purpose
Server and client random	Byte sequences for establishing connection
Server write MAC secret	The secret used in MAC operations on data written by the server
Client write MAC secret	The secret used in MAC operations on data written by the client
Server write key	The bulk cipher key for data encrypted by the server and decrypted by the client
Client write key	The bulk cipher key for data encrypted by the client and decrypted by the server
Initialization vectors	Random number to initialize encryption operation
Sequence numbers	Sequence numbers for transmitted and received messages for each connection

b. SSL Record Layer Protocol

✍ ***Definition :*** *The SSL Record Layer is the last protocol that receives the raw data from the higher application layers and other SSL protocols such as handshake.*

- Its core function is to facilitate (perform) data transfer. The basic unit of data in SSL is a record. Each record consists of a five-byte record header, followed by data.
- There are four types of records in SSL.

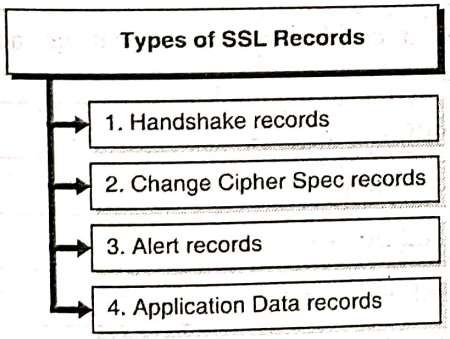


Fig. 6.2.2 : Types of records in SSL

- The five-byte format of an SSL Record Header is as follows:

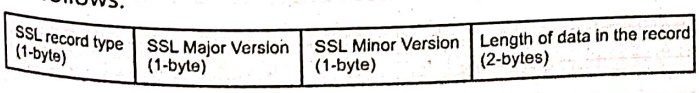


Fig. 6.2.3

- Following is a simplistic block diagram of steps involved in the SSL Record Protocol.

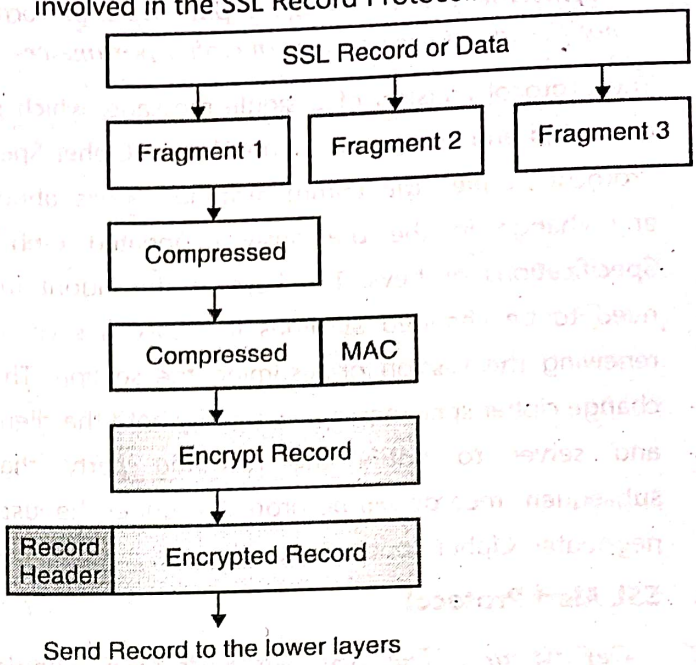


Fig. 6.2.4 : Steps in the SSL record protocol

- At a high level the SSL Record Protocol performs three operations :

Operation Performed	Purpose
Fragmentation	Break original data into SSL Plaintext records of 2^{14} bytes or less
Compression and Decompression	All SSL Plaintext records are compressed using the compression algorithm defined in the current session state. The compression algorithm translates an SSL Plaintext structure into an SSL Compressed structure
Payload Protection	All SSL Compressed records are protected using the encryption and MAC algorithms defined in the current Cipher Spec. Once the handshake is complete, the two parties have shared secrets that are used to encrypt records and compute keyed Message Authentication Codes (MACs) on their contents. The techniques used to perform the encryption and MAC operations are defined by the Cipher Spec.

1. SSL Change Cipher Spec Protocol

✍ ***Definition :*** *The Change Cipher Spec protocol notifies about the changes in cipher parameters.*

The protocol consists of a single message, which is encrypted and compressed. The Change Cipher Spec Protocol notifies the communicating parties about any change in the previously negotiated Cipher Specifications or Keys. The keys or the algorithms need to be changed at times for reasons such as renewing the session or resuming the session. The change cipher spec message is sent by both the client and server to notify the receiving party that subsequent records will be protected under the just-negotiated Cipher Spec and keys.

2. SSL Alert Protocol

✍ ***Definition :*** *The SSL Alert Protocol signals problems with an SSL session.*

- One of the content types supported by the SSL record layer is the alert type.
- Alert messages notify the
  - o severity of the alert and
  - o a description of the alert
- Alert messages with a severity level of *fatal* result in the immediate termination of the connection. In this case, other connections corresponding to the session may continue, but the session identifier is invalidated, preventing the failed session from being used to establish new connections. Like other messages, alert messages are encrypted and compressed, as specified by the current connection state.
- The following Table 6.2.1 summarizes the various alert records.

Table 6.2.1

Alert Code	Alert Message	Alert Level	Alert Description
0	close_notify	1 (Warning)	notifies the recipient that the sender will not send any more messages on this connection
10	unexpected_message	2 (Fatal)	An inappropriate message was received
20	bad_record_mac	2 (Fatal)	A record is received with an incorrect MAC
30	decompression_failure	2 (Fatal)	The decompression function received improper input
40	handshake_failure	2 (Fatal)	The sender was unable to negotiate an acceptable set of security parameters given the options available
41	no_certificate	1 (Warning)	Sent in response to a certification request if no appropriate certificate is available
42	bad_certificate	1 (Warning)	A certificate was corrupt
43	unsupported_certificate	1 (Warning)	A certificate was of an unsupported type
44	certificate_revoked	1 (Warning)	A certificate was revoked by its signer
45	certificate_expired	1 (Warning)	A certificate has expired or is not currently valid
46	certificate_unknown	1 (Warning)	Some other (unspecified) issues
47	illegal_parameter	2 (Fatal)	A field in the handshake was out of range or inconsistent with other fields

- The alert record consists of 2 bytes of information from the table above.

Alert Level (1-byte)	Alert Code (1-byte)

3. SSL Handshake Protocol

✍ ***Definition :*** *The cryptographic parameters of the session state are produced by the SSL handshake protocol.*

- When an SSL client and the server first start communicating, they need to agree upon certain parameters. There are also several steps that need to be carried out to establish a secure session. At a high level, the following four steps are carried out.

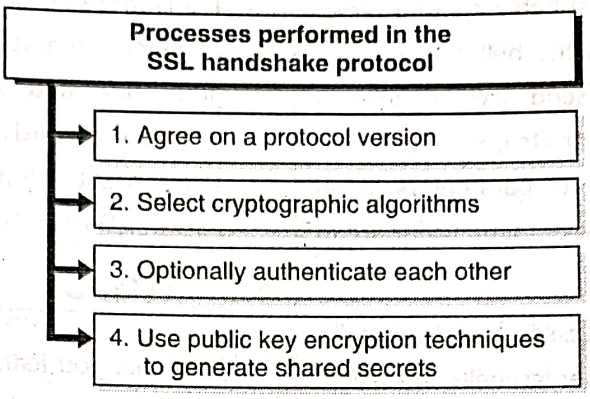


Fig. 6.2.5

- The steps can be detailed in the following simplistic handshake process diagram.

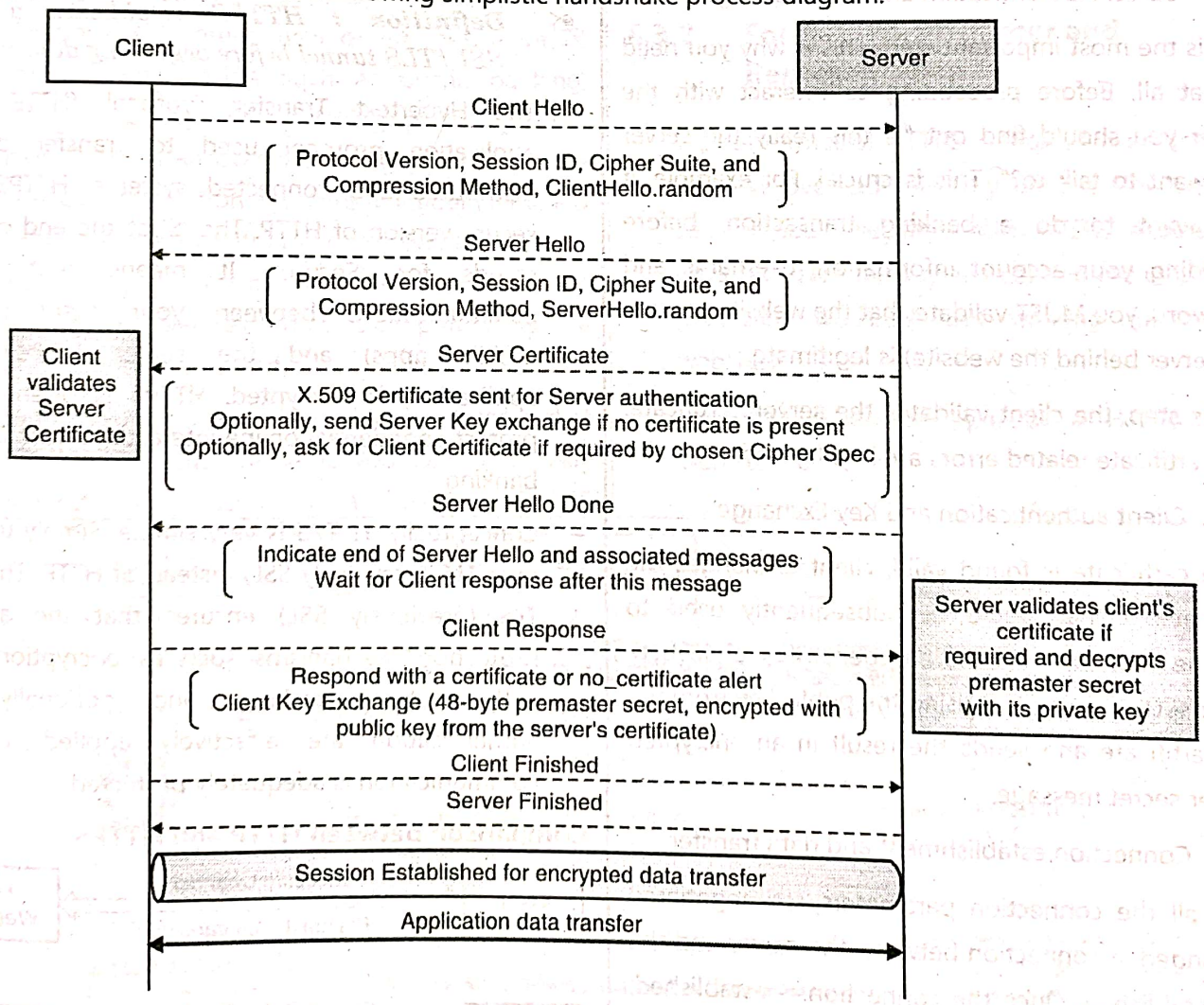


Fig. 6.2.6

Step 1 : Hello messages (Establish security capabilities)

The hello phase messages are used to exchange security enhancement capabilities between the client and server.

1. **Client Hello :** When a client first connects to a server it is required to send the client hello as its first message. The client can also send a client hello in response to a hello request or on its own initiative in order to renegotiate the security parameters in an existing connection. The list of parameters sent are in the diagram.
2. **Server Hello :** The server processes the client hello message and responds with server hello message. The list of parameters sent are in the diagram.

Step 2 : Server Authentication and Key Exchange

- This is the most important step. This is why you need SSL at all. Before proceeding to interact with the server you should find out "Is this really the server you want to talk to?". This is crucial. For example, if you want to do a banking transaction, before providing your account information, username and password, you MUST validate that the website you are on (server behind the website) is legitimate.
- In this step, the client validates the server certificate. Any certificate related errors are highlighted.

Step 3 : Client authentication and Key Exchange

If the certificate is found valid, client exchanges the keying material that would be subsequently used to encrypt the messages. The client generates a 48-byte premaster secret, encrypts it using the public key from the server's certificate and sends the result in an encrypted pre-master secret message.

Step 4 : Connection establishment and data transfer

Once all the connection parameters are negotiated and exchanged, a connection between the server and the client is established. Once the connection is established, the data transfer begins between the server and the client. The data is encrypted based on the negotiated terms.

6.2.3 Transport Layer Security (TLS)

- As you learnt earlier, SSL is obsolete. TLS replaced SSL in 1999. The underlying working of TLS is very similar to SSL.
- TLS is more efficient and secure than SSL. It provides stronger message authentication, key-material generation and supports pre-shared keys, secure remote passwords, elliptical-curve keys and Kerberos. TLS and SSL are not interoperable, but TLS provides backward compatibility for devices using SSL.

6.3 HTTPS

- Now that you learnt how SSL works, let's learn about one of its implementations – HTTPS application protocol.

✍ ***Definition** : HTTPS establishes a secure SSL / TLS tunnel before beginning data transfer.*

- The Hypertext Transfer Protocol (HTTP) is an application protocol used to transfer data on distributed and connected systems. HTTPS is the secure version of HTTP. The 'S' at the end of HTTPS stands for 'Secure'. It means that all the communications between your client (browser, mobile apps) and the server (website, web application) is encrypted. HTTPS is often used to protect confidential online interactions such as online banking.
- Conceptually, HTTPS is very simple. Simply use HTTP over TLS (previously SSL) instead of HTTP. The use of TLS (previously SSL) ensures that the adequate protection mechanisms such as encryption, server authentication, hashing, and optionally client authentication are effectively applied, and the communication is adequately protected.

Comparison between HTTP and HTTPS

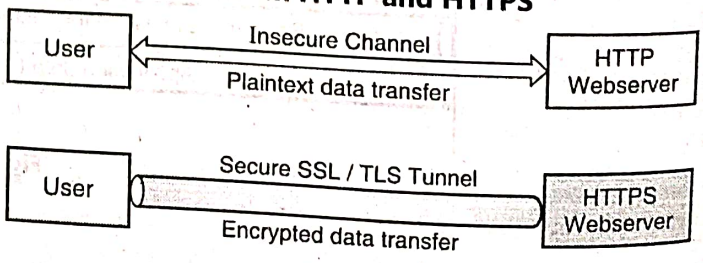


Fig. 6.3.1

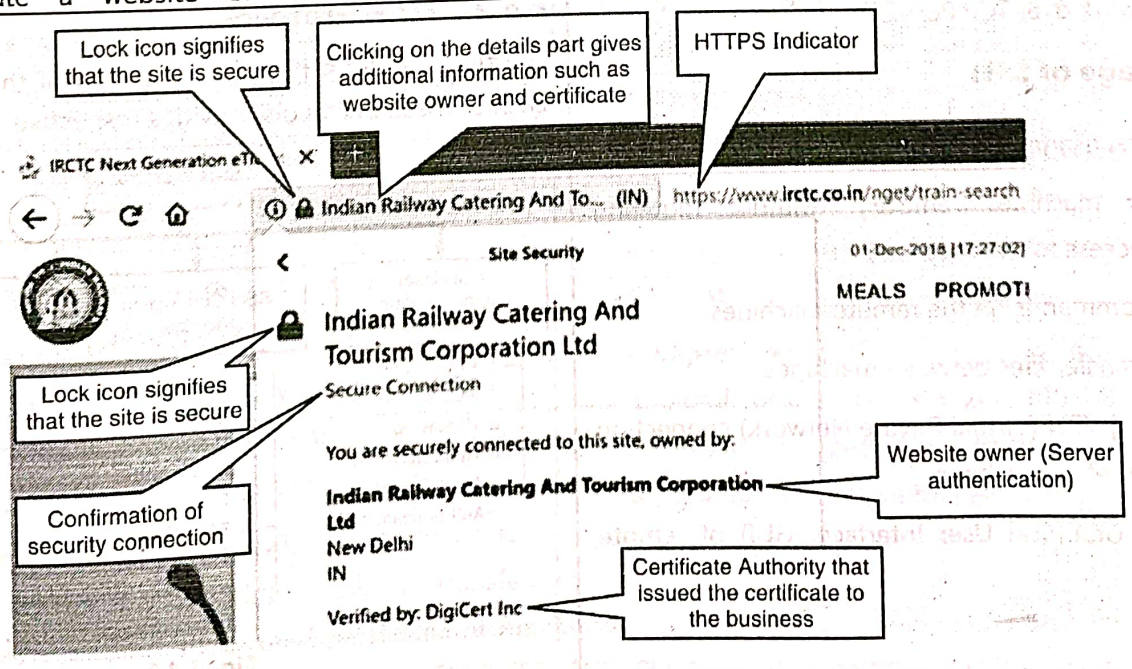
HTTP	HTTPS
Data transfer in plaintext	Data transfer in ciphertext
Default port is 80	Default port is 443
Does not require SSL/TLS or Certificates	Requires SSL/TLS implementation with Certificates
URL has http://	URL has https://
Should be avoided	Should be preferred
Search engines do not favour the insecure websites	Improved reputation of the website in search engine
Users worried about their data	Users confident about the security of their data

6.3.1 Motivation / Benefits of using HTTPS

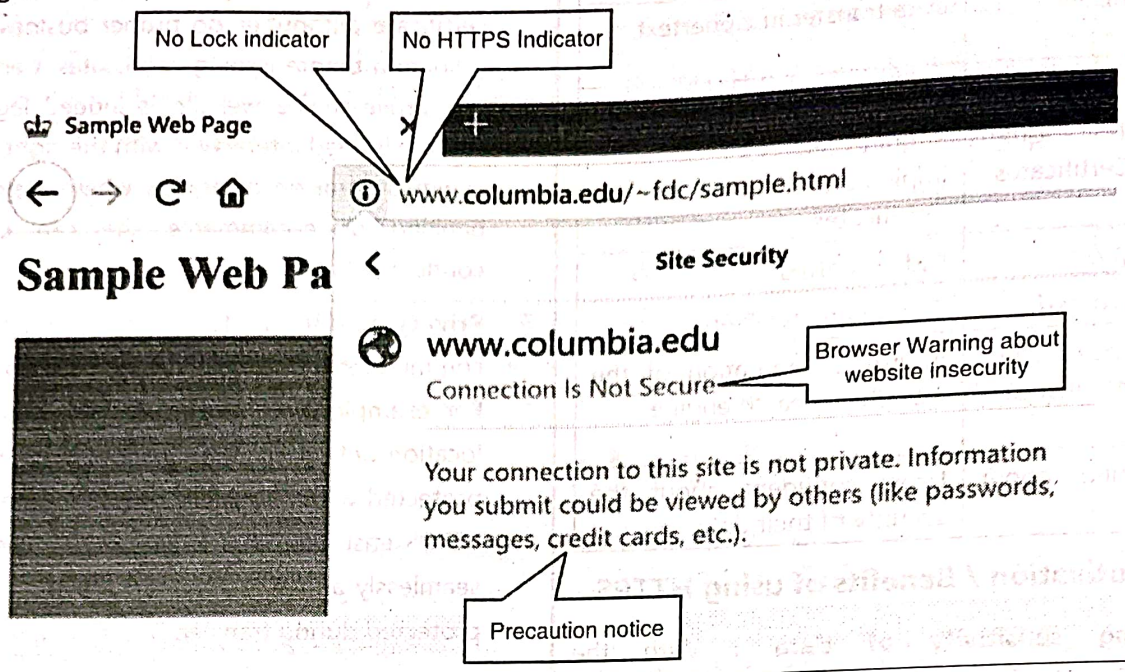
1. **Increasing sensitivity of data :** With the proliferation (widespread use) of internet, a lot of sensitive communication such as online banking, ticketing, shopping, etc. is taking place over the Internet. There is an ever increasing need to ensure that the communication is secure (confidentiality and integrity of the information is enforced). Information such as your password or credit card number is not transferred in plaintext that can potentially be captured and then misused.
2. **Authentication :** One of the critical use cases that HTTPS serves is that using it you can potentially authenticate a website or a business. HTTPS connection is established using X.509 certificates and certificate authorities do proper business or website validation before issuing certificates. Certificates help you prove that a website is indeed legitimate, and you are indeed interacting with the right website. This avoids several online frauds where a similar looking banking or e-commerce site can capture your confidential details.
3. **Privacy requirements :** Often times, the nature of communication is private even if it is not confidential. For example, your health reports, your chats, your location details, etc. require that they are adequately protected when transferred over the network. Use of HTTPS ensures that encryption is applied to all data seamlessly and the private information is adequately protected during transfer.

6.3.2 Format, Port Number and Representation

- Typical format of HTTPS is https://www.example.com. It works over port 443 by default. You would have seen various websites with HTTPS enabled. These days browsers show green color in the URL for HTTPS protected websites and warning for non-HTTPS websites.
- Following is an example of a HTTPS protected website.



– Following is an example of a non-HTTPS website.



6.4 Secure Shell (SSH)

✍ ***Definition :*** *Secure Shell (SSH) is a protocol for secure remote login and other secure network services over an insecure network.*

– The SSH protocol was originally developed by Tatu Ylonen in 1995. The objective behind the development of SSH was to secure network connections and obsolete insecure protocols such as telnet, rlogin, rsh, rexec with their secure counterparts implemented over the SSH protocol. SSH servers typically work over TCP Port 22.

6.4.1 Usage of SSH

SSH has several usages. Some of them are as follows.

1. Login to machines remotely (without requiring physical access to the machine)
2. Execute commands on the remote machines
3. Copy or transfer files between machines
4. Establishing VPN (Virtual Private Network) connection between a set of machines
5. Accessing Graphical User Interface (GUI) of remote machines
6. Secure communication between machines
7. Compress data before transfer

Security services provided by SSH

SSH provides several security services. Some of them are as follows.

1. Confidentiality via encryption
2. Integrity via hashing
3. Server Authentication
4. Client User Authentication

6.4.2 SSH Protocol

The overall SSH protocol consists of three protocols. Each of these protocols provides respective services.

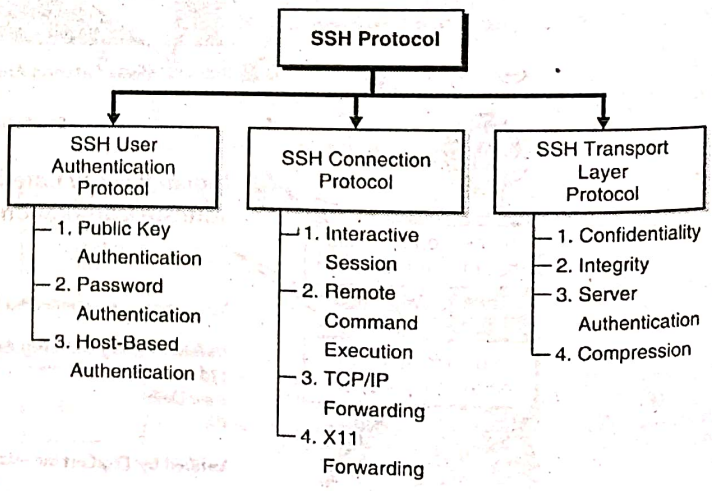


Fig. 6.4.1

1. SSH Authentication Protocol

✍ ***Definition :*** *SSH Authentication Protocol performs client user authentication.*

Whenever an SSH client tries to establish a connection with the SSH server, the server requires that the client authenticates with the SSH user that must be present on the SSH server. A client cannot establish an SSH connection without an SSH user already available on the SSH server. The authentication protocol ensures that the user can successfully authenticate with the SSH server. The protocol provides three mechanisms for carrying out the user authentication.

1. **Public Key Authentication :** The client signs the message (using client's private key) containing the client's public key and sends it to the server. The server validates the signature and the underlying public key received and allows access if the signature is correct.
2. **Password Authentication :** The client sends a password to the server for authentication. The server validates the password and allows access if the provided password is correct.
3. **Host-Based Authentication :** In this authentication mechanism, the authentication process is carried out by the host (machine) where the client is installed. The server validates the host instead of the client.

2. SSH Connection Protocol

✍ ***Definition :*** *The SSH Connection Protocol deals with managing SSH connections between the client and the server.*

It provides interactive login sessions, remote execution of commands, forwarding TCP/IP connections, and forwarding X11 connections.

1. **Interactive session :** It opens a working session. In a session, you could execute commands, or carry out administrative tasks on the remote machines.
2. **Remote execution of commands :** Once the session has been set up, any program (shell, file transfer, email, document editor, etc.) could be started at the remote machine. You could then execute various commands or carry out activities as desired.
3. **Forwarding TCP/IP connections :** Forwarding provides the ability to carry out any insecure TCP connection over a secure SSH connection. The insecure TCP connection passes through the secure SSH tunnel and hence the insecure connection automatically gets the security from the established SSH tunnel. This is also referred to as SSH tunneling.
4. **Forwarding X11 connections :** X11 forwarding allows you to run any GUI application on a remote machine.

3. SSH Transport Layer Protocol

✍ ***Definition :*** *The SSH transport layer is a secure, low level transport protocol. It provides strong encryption, cryptographic host authentication, and integrity protection.*

The Transport Layer Protocol provides server authentication, confidentiality, and integrity. It may optionally also provide compression. The transport layer typically runs over a TCP/IP connection, but might also be used on top of any other reliable data stream.

1. **Confidentiality :** Confidentiality is provided via the negotiated encryption algorithms.
2. **Integrity :** Data integrity is protected by including with each packet a MAC that is computed from a shared secret, packet sequence number, and the contents of the packet.
3. **Server authentication :** The SSH server host key is used during key exchange [when client tries to connect to the server] to authenticate the identity of the host.
4. **Compression :** If compression is required, the 'payload' field is compressed using the negotiated algorithm. The 'packet_length' field and the 'mac' is computed from the compressed payload. Encryption is done after compression.

Service	Algorithm
Confidentiality (Encryption)	3des-cbc blowfish-cbc twofish256-cbc twofish-cbc twofish192-cbc twofish128-cbc aes256-cbc aes192-cbc aes128-cbc serpent256-cbc serpent192-cbc serpent128-cbc arcfour idea-cbc cast128-cbc none (no encryption)
Integrity (Hashing)	hmac-sha1 hmac-sha1-96 hmac-md5 hmac-md5-96 none (no hashing)
Compression	zlib none (no compression)

The various algorithms supported at the SSH transport layer are as follows.

6.4.3 Establishing SSH connection

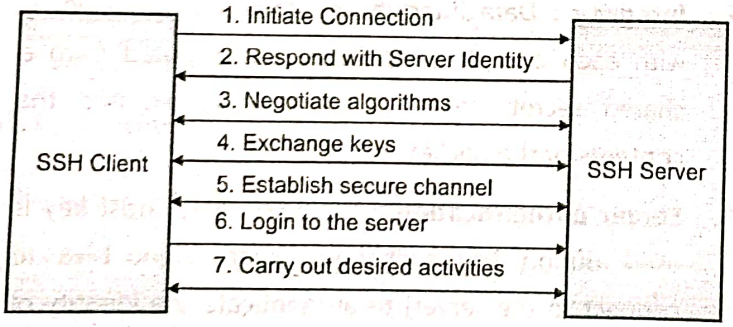


Fig. 6.4.2

This is a highly simplistic SSH connection diagram. The client initiates the connection and a secure channel is established after verifying server identity, negotiating algorithm parameters and exchanging keys. Once the channel is established, the client provides the user authentication after which the remote activities can begin.

6.5 Secure Electronic Transactions (SET)

✍ ***Definition :*** *Secure Electronic Transaction (SET) is a security technology proposed by Visa and MasterCard to allow for secure credit card transaction.*

- Although SET provides an effective way of transmitting credit card information, it is not used in the industry today due to various complexities involved. The world has accepted SSL as the preferred way of conducting secure transaction online. The HTTPS (HTTP over SSL) provides a secure way to carry out online transactions.
- SET is a cryptographic protocol and infrastructure developed to send encrypted credit card numbers over the Internet. The following entities are involved in a SET-based transaction.

1. **Issuer (cardholder's bank) :** The financial institution that provides a credit card to the individual.
2. **User (or the Cardholder) :** The individual authorised to use a credit card.
3. **Merchant (or the vendor) :** The organisation or entity providing goods or services.
4. **Acquirer (merchant's bank) :** The financial institution that processes payment.
5. **Payment gateway :** Entity that initializes and approves the payment.

Steps involved in a SET-based transaction

Note : Both cardholders and merchants must register with CA (Certificate Authority) first, before they can buy or sell on the Internet.

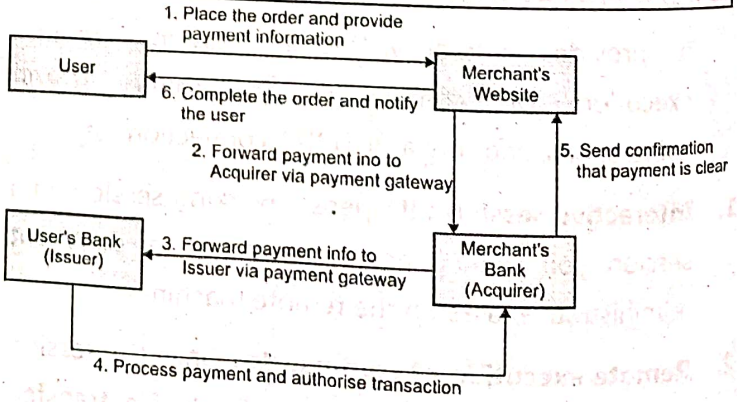


Fig. 6.5.1

1. User browses the merchant's website and decides the order. She sends the order and payment information
   (a) **Purchase Order :** this part is for merchant
   (b) **Card Information :** this part is for merchant's bank only (acquirer)
2. Merchant forwards the card information to its bank via the payment gateway
3. Merchant's bank sends the credit card information to the Issuer via the payment gateway
4. Issuer verifies the credit card information and sends the authorisation to the Merchant's bank
5. Merchant's bank send authorisation to merchant notifying that the payment is cleared
6. Merchant completes the order and sends confirmation to the customer

6.6 User Authentication and Session Management

- HTTP is a stateless protocol. Each HTTP request is independent of the previous interactions. Now, imagine an experience. You login to a secure banking or e-commerce website and for every click and access to any resource on the webpage you need to authenticate again. Would you like it? How many times will you have to authenticate before you can complete a transaction?
- Now think from the webserver perspective : how should it determine that you are still authenticated, and it can proceed with allowing you to access the webpage resource? What if it made a mistake in correctly identifying you and allowed someone else to carry out the transaction using your account details?
- This is precisely the problem you can solve using sessions.

✍ ***Definition :*** *Session is a mechanism using which a webserver (or an application) can save the state of various aspects with respect to the user engagement on the website or application.*

- What does that mean? Broadly speaking, a session can hold several information such as
  - User authentication
  - User authorisation
  - Which parts of websites you have visited
  - What is in your cart
- Webservers or applications require retaining the information or status about each user for the duration of multiple requests. Sessions provide the ability to retain such information. The information saved in the session applies to each and every interaction the user has with the web application for the duration of the session. So, whether you are authenticated or not is saved in the session state and the webpage does not ask you to authenticate on every click. Relax now!

6.6.1 Session Bindings

- A session is created between the two parties – usually the software running at the user end (browser, OS or an app) and the webserver (application or the session host). A session secret is created and shared between the two parties that allows them to link and refer to the session and continue and allow the requests. This secret is shared (or cryptographically proven to be present) by the user's software every time it requires accessing a resource.
- The session secret typically has the following characteristics.
  1. It should be generated by the session host (webserver or application) after a successful authentication event.
  2. It should be randomly generated.
  3. It should be erased when the session is terminated.
  4. It should be protected when stored and should not be transmitted in cleartext.
  5. It should have an appropriate lifetime and should not be possible to use after the expiry.
- The session secret that is typically used is Session Identifier (Session ID).

6.6.2 Session Lifecycle

Following is a typical session lifecycle.

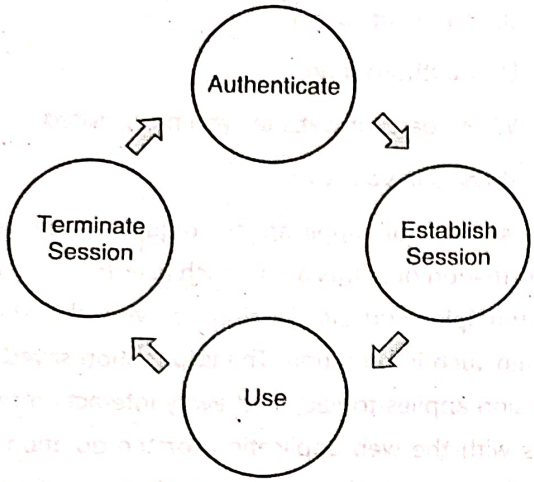


Fig. 6.6.1

1. **Authenticate :** The user begins to establish a session by authenticating to the webserver or the application that she is trying to access. This could be any authentication mechanism that the user has signed up for before such as username and password or biometric.
2. **Establish Session :** Once the user is successfully authenticated, the session host (webserver or application) creates a session secret (usually a session ID) and shares it with the client (browser, OS or the app). The session can have several other information such as user language preference, location or anything else that the session host requires to deliver a good user experience.
3. **Use :** The user can then make several requests to the session host as long as the session is active and can access the resources as per the authorization she has. On each request, the client software attaches the session secret (session ID) that was established in the previous step.
4. **Terminate Session :** Once the user is done with all the required interactions and has no further requests to make, then she can terminate the session by logging out. The session can also terminate by itself due to inactivity to avoid allowing someone else to hijack an unused session.

6.6.3 Session Management Mechanisms

Mostly there are three mechanisms to manage sessions.

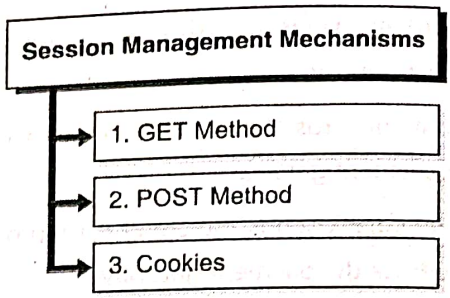


Fig. 6.6.2 : Session management mechanisms

1. **GET Method :** GET method encodes the session data as part of the URL. The Session ID is put in the URL itself.

   For example, http://example.com/user.php?SessionID=2aqw45763 8487438. The webserver checks the Session ID to be valid. If it is found to be valid, the request is served. This method is also called URL Rewriting.
2. **POST Method :** POST method is a way to send data to the webserver. The Session ID can be sent along with the request details. The user does not see the Session ID information in the URL. This method is also called HTML Hidden Field.
3. **Cookies :** Using HTTP cookies or simply cookies is the predominant way of maintaining sessions. It provides several security benefits that the other two mechanisms do not. Cookie is a file containing the information required to save the session state or settings for a website.

Comparison between Session Management Mechanisms

Comparison Attribute	GET Method	POST Method	Cookies
Session ID in URL	Yes	No	No
Security	Medium	Medium	High
Predominant use	No	No	Yes

6.6.3(A) Cookies

Definition : *Cookie is a piece of state information supplied by a webserver to a browser, in a response for a requested resource, for the browser to store temporarily and return to the server on any subsequent visits or requests.*

- Cookies are used for session management and other purposes.
- Note here that the cookies can store a variety of information and have several other purposes than session management such as user experience personalisation. I will restrict our discussion to only cookies with respect to their security and their use in session management.

Structure and Content of Cookies

Cookies are set by the webserver using the `Set-Cookie` HTTP response header.

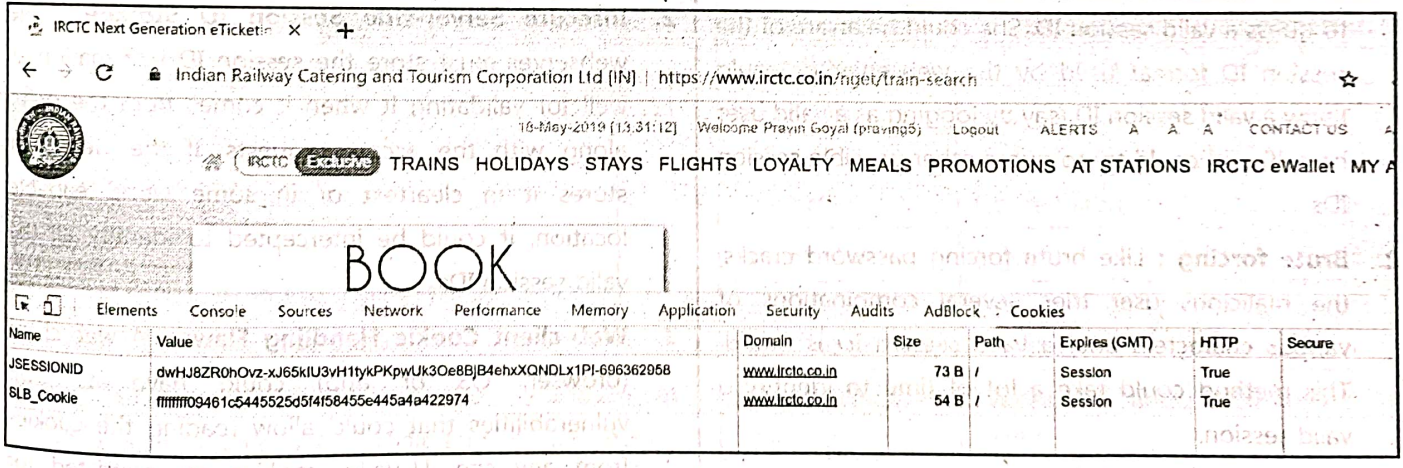
For example,

```
Set-Cookie: sessionToken=12345678; Expires=Sat, 18 May 2019 10:18:14 GMT
```

- Cookies allow the webserver to save various information at the user end. Cookies also have the following parameters that the webserver can define to manage the cookies.

Cookie Parameter	Purpose
Name	Name of the cookie parameter that the webserver wants to set e.g. SessionID
Value	Value of the cookies parameter that the webserver wants to set e.g. 12345678
Expires	Date of cookie expiry. If not set, cookie will be deleted when the browser is closed. It is recommended to set this parameter to minimum duration as appropriate.
Max-age	Alternate way to define expiry in seconds from the time cookie was received
Domain	The network domain that the cookie applies to e.g. example.com
Path	The webpage path that the cookie applies to e.g. /user
Secure	Indicates that the cookie can only be used with https. It is recommended to set this.
HttpOnly	Indicates that the cookie can only be modified and used by the webserver. Clients cannot access the cookie. It is recommended to set this.

- For example, look at the cookie that gets set when I login to https://www.irctc.co.in



Can you match with the table of information that I provided earlier?

- You can see that the cookie information is not set to `Secure`. It means that this cookie can be used with both http and https. You should not do this for your web application until it is required.

- You also see that the `HTTP` field is set to `True`. That is for the `HttpOnly` parameter. It represents that this cookie can only be accessed and modified by the webserver. You should follow this.
- You also see that the cookie expiry is set to session only. Once the session is expired, the cookie would expire as well. This is also a recommended practice.

6.6.4 Attacks on Session Management

- Attacks on session management focus on retrieving a valid session ID. A valid session ID can allow the malicious user to take over your session and carry out activities on your behalf.
- Primarily there are two types of attacks on session management.

1. Session Hijacking
2. Session Fixation

Let's learn about each of them in detail.

1. Session Hijacking

✍ ***Definition :*** *Session hijacking is an attack where a malicious user can acquire a valid session.*

The malicious user can identify a valid session using

1. **Prediction :** In this method, the user randomly tries to guess a valid session ID. She could be aware of the session ID format used by the webserver or could know a valid session ID (say by logging as a valid user herself) and could try to guess other possible session IDs.
2. **Brute forcing :** Like brute forcing password cracks, the malicious user tries several combinations of various characters until a valid session ID is found. This method could take a lot of time to identity a valid session.
3. **Interception :** Interception is the most common method of getting a valid session ID because it directly attempts to sniff a valid session from the active user.

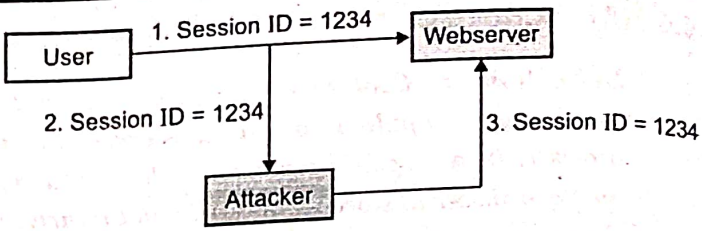


Fig. 6.6.3

There could be various interception techniques or vulnerabilities that allow intercepting a valid session ID. Let's learn about them.

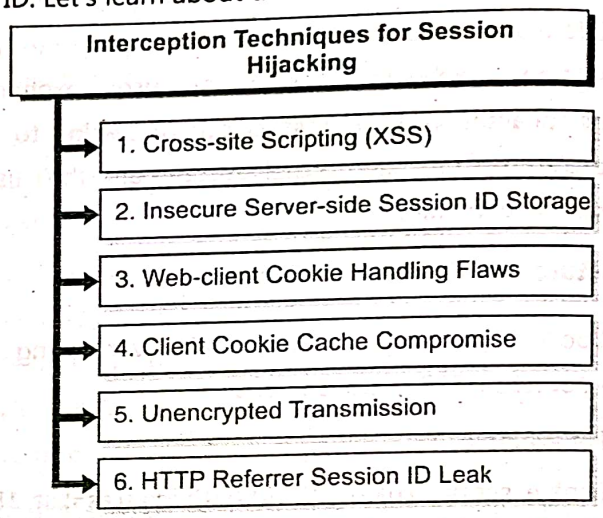


Fig. 6.6.4

1. **Cross-site Scripting (XSS) :** You have already learnt about XSS in Unit 2. Please refer that section for a quick understanding and revision. Using XSS, you can launch attacks that can reveal valid session IDs.
2. **Insecure Server-side Session ID Storage :** The webserver must store the session ID information as well for validating it when it comes from the client along with the access requests. If the webserver stores it in cleartext or in some world-readable location, it could be intercepted to identity all the valid session IDs.
3. **Web-client Cookie Handling Flaws :** A web-client (browser, OS or app) could have its own vulnerabilities that could allow reading the cookies from any site. Usually, cookies are restricted for access only by the domain and for the path specified in the cookies and should not be allowed access from any other site.

4. **Client Cookie Cache Compromise :** Cookies that are long lived (with long expiry dates) are usually stored in cookie cache location. If an attacker has already compromised a machine, she could access the cookie cache to get the session IDs and other sensitive information.

5. **Unencrypted Transmission :** If you are not using a https connection, then the cookies are transferred in cleartext on the http connection. A malicious user can then intercept such traffic and get the session ID. It is important to mark cookies as secure so that they are transferred only on a https connection. The attacker should not be able to force a http connection and get the cookies in cleartext.

6. **HTTP Referrer Session ID Leak :** HTTP has a HTTP Referrer attribute that tracks what page has the client come from to the current location. The referrer name contains the entire URL of the previous page and could include session IDs if the GET method is used for session management. This HTTP Referrer line could expose the session ID to sites that are visited immediately after leaving a secured website page.

2. Session Fixation

Definition : Session fixation is an attack where the attacker can force a client to use a specific session ID.

Let's take a simple example.

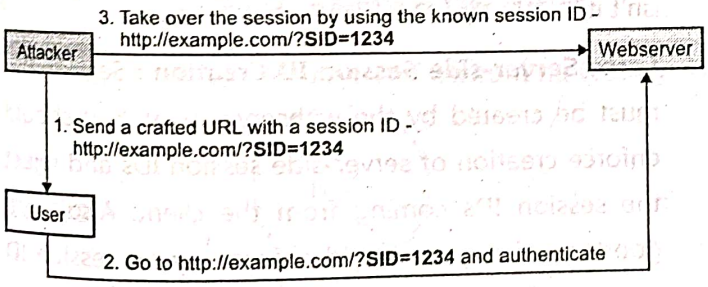


Fig. 6.6.5

1. The attacker identities that the webserver at http://example.com is vulnerable and accepts session IDs from the client. She creates a session ID (SID 1234) and sends the user an email with the link containing the session ID.
2. The unsuspicious user clicks on the link and goes to http://example.com website with the session ID provided by the attacker (SID 1234). The user authenticates to the website when asked.
3. The attacker is already aware of the session ID as she created it. She then takes over the session and could control activities on behalf of the user.

– This is a basic webserver vulnerability where the webserver does not enforce the generation of a new session ID immediately after a successful authentication. Typically, the webservers assign a session ID only once during an active session. So, when they see that the client already has a valid session ID, they do not assign a new one. Now, if the webserver has not enforced creating and using a new session ID mandatorily after a successful authentication attempt, it can continue to use the session ID that the client sent. The attacker can then potentially take advantage of this vulnerability by creating a session ID and forcing the user to use the session ID fixed by the attacker.

– As a countermeasure, the webserver must always create a new session ID immediately after a successful authentication.

Comparison between Session Hijacking and Session Fixation

Comparison Attribute	Session Hijacking	Session Fixation
Session ID	Not known before hand	Known before hand
User action	Not required	Required
Complexity	High	Low
Mechanism Exploited (mostly)	GET, POST, Cookies	GET and POST
Vulnerability at	Webserver and Client	Webserver

6.6.5 Countermeasures for Preventing Attacks on Sessions

Following are some of the recommended best practices for preventing attacks on sessions.

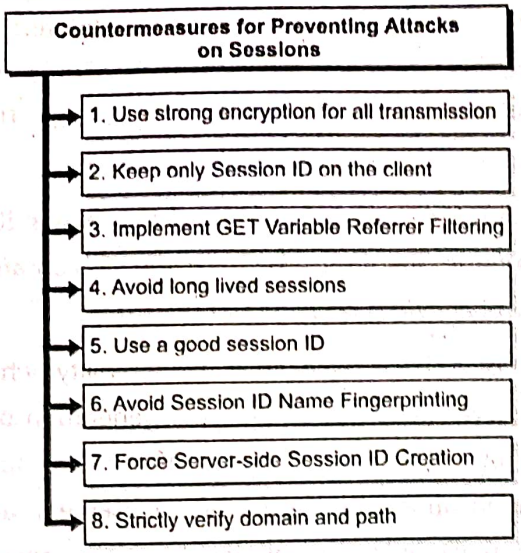


Fig. 6.6.6

1. **Use strong encryption for all transmission :** You should use a TLS connection with strong ciphers for all communication. Ensure that the cookies are set to Secure and HttpOnly. If you are using GET or POST method for session management, ensure that you use TLS as well.
2. **Keep only Session ID on the client :** Do not store sensitive information on the client side. You should mostly just keep the session ID on the client side and all the sensitive information should be stored on the server side. The sensitive information on client side could be manipulated.
3. **Implement GET Variable Referrer Filtering :** If you are using GET method for session management, then implement HTTP referrer filtering where the session ID is filtered out when the user is sent to the different site.
4. **Avoid long lived sessions** : You should expire the sessions that are inactive. You should also set an appropriate expiry time for cookies. Also, as a good practice, you can frequently change the session IDs for the active sessions to make it harder for the attacker to take over the sessions.
5. **Use a good session ID :** The session ID should be difficult to predict or guess. You should create a session ID using a random function generator or by enforcing other forms of randomness. Also, ensure that the session ID is long enough and is not repeated.
6. **Avoid Session ID Name Fingerprinting**

- The name used by the session ID should not be extremely descriptive. It should not disclose unnecessary details about the purpose and meaning of the ID. The session ID names used by the most common web application development frameworks can be easily fingerprinted (identified uniquely).
- For example
  - PHP uses `PHPSESSID` by default.
  - Java uses `JSESSIONID` by default.
  - ColdFusion uses `CFID` and `CFTOKEN` by default.
  - ASP.NET uses `ASP.NET_SessionId` by default.
- Therefore, the session ID name can disclose the technologies and programming languages used by the web application. This can make the task of the attacker easier and she can try to exploit known vulnerabilities in those programming languages or application development frameworks. It is recommended to change the default session ID name of the web development framework to a generic name, such as id.
- By the way, do you recall where did you see `JSESSIONID` being used as the session ID variable name? Go back and check the cookies section where I have given the irctc website example. So, now you know that the irctc application is Java based! Cool, isn't it?

7. **Force Server-side Session ID Creation :** Session IDs must be created by the webserver only. You should enforce creation of server-side session IDs and reject the session IDs coming from the client. Also, as a good practice, you should enforce a new session ID creation after a successful authentication and discard the previous ones, if any.
8. **Strictly verify domain and path :** You should strictly verify the domain and the path that a cookie is referring to. Before accepting the cookie, ensure that the cookie was issued from the right domain and for the right page (path).

6.7 Web Browser Attacks

There are several attacks that could be potentially carried out on the web. You have already seen quite a few so far in this book. Let's learn a few more in this section.

6.7.1 Account Harvesting

Definition : Account harvesting is the process of gathering user accounts from a system, service, or database.

- Attackers can use several methods to gather account related information. The typical account related information gathered is
  - Username
  - Email address
  - Phone Number
  - Account Privileges
- The gathered information can then be used to send spam, try to login into systems using the account information or carry out general fraudulent activities.
- Note here that the term account harvesting can also be referred by other terms in the industry such as credential harvesting, email harvesting, username harvesting, etc. They all mean the same and are purposed to gather as much account related information as possible.

6.7.1(A) Account Harvesting Lifecycle

Following is a typical lifecycle of account harvesting.

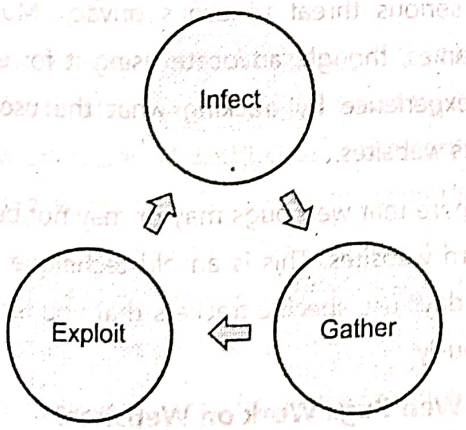


Fig. 6.7.1

1. **Infect :** An attacker can spread infection in several ways –
   - Injection attacks
   - Website hijacking or cloning
   - Phishing
   - Social engineering
   - Malware
   - Other malicious activities

   Once the user comes in contact with such an infection, she unknowingly falls victim to the attacker.

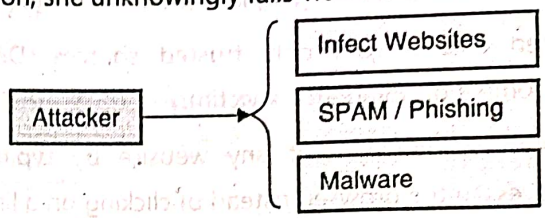


Fig. 6.7.2

2. **Gather**
   - Based on the infection the user comes in contact with, the account information is gathered accordingly.
   - For example, if the user clicks on the phishing link, it may open her bank look-alike website and she may provide her account details there. The attacker typically redirects the user to the original website after collecting the account information so that the user perceives that she might have put wrong information first and hence the website didn't allow her to login and the page refreshed to provide her the second attempt.
   - Similarly, the attacker can also gather account information by infecting the websites. Users visiting the infected websites, may login and then automatically provide their information to the attacker.
   - Apart from this, the attacker might infect the device by putting a malware. The malware can then send account related information to the attacker by collecting the various account details as and when the user uses those details to login to various websites.
3. **Exploit :** Once the attacker has the required account information, she can then begin exploiting that information. The information exploitation could lead

to account hijacking (take over) or the attacker can use the account for further spreading infection (SPAM, phishing, etc.). It might be very difficult for the user to regain control over his account once it is taken over.

6.7.1(B) Countermeasures against Account Harvesting

- Following are some of the recommended best practices for protecting against account harvesting.
- Open emails from only trusted sources. Do not become a phishing attack victim.
- Prefer to directly visit any website by typing its address in the browser instead of clicking on a link.
- Be selective in which websites you go to and what information you share publicly.
- Create and use throw away accounts for websites that just need that information for marketing and spamming without providing any additional value based on your account details.
- Use multi-factor authentication to protect your account from login with just username and password.
- Protect and secure your devices.

6.7.2 Web Bugs

✍ ***Definition :*** *Web bug is a tiny image, invisible to a user, placed on web pages to track the users and collect information.*

> **Note :** Don't get confused with the term "bug" in web bugs. Here I am not talking about the regular software bugs (flaws) in the programs, websites or applications. Web bug is more specifically a tracking technique. By the way, generally speaking, bugs are microorganisms that form a species of insects and may cause illness.

- Web bug could be considered a type of malicious code that allows third-parties to collect various user information. Typically the following information is collected.
  - User account details
  - IP address
  - Hostname (User's computer name)
  - Browser type and version
  - Operating System Name and Version
  - Cookies
- Web bugs are also commonly referred as
  - Web beacons
  - Tracking bug
  - Tag
  - Web tag
  - Page tag
  - Tracking pixel
  - Pixel tag
  - 1×1 GIF
  - Clear GIF
- Web bug is a very tiny image (usually 1x1 pixel sized) that is transparent or so small that the human eye cannot see it. However, a web bug can also be visible. Whether it is visible or not visible does not make any difference to its purpose and the way it works.
- Web bugs can be placed in WebPages as well as html based emails. In emails, web bugs can be used to validate an email address or know if the user has opened an email apart from the regular information collection (IP Address, software using which email was read or the time of email reading). Web bugs could be a serious threat to user's privacy. Most of the companies, though, advocate using it for enhancing user experience by tracking what the users do on various websites.
- Note here that web bugs may or may not be used on modern websites. This is an old technique. Websites these days use specific trackers that you learnt about previously.

(A) How Web Bugs Work on Websites?

- Following is a typical layout of a web bug on websites.

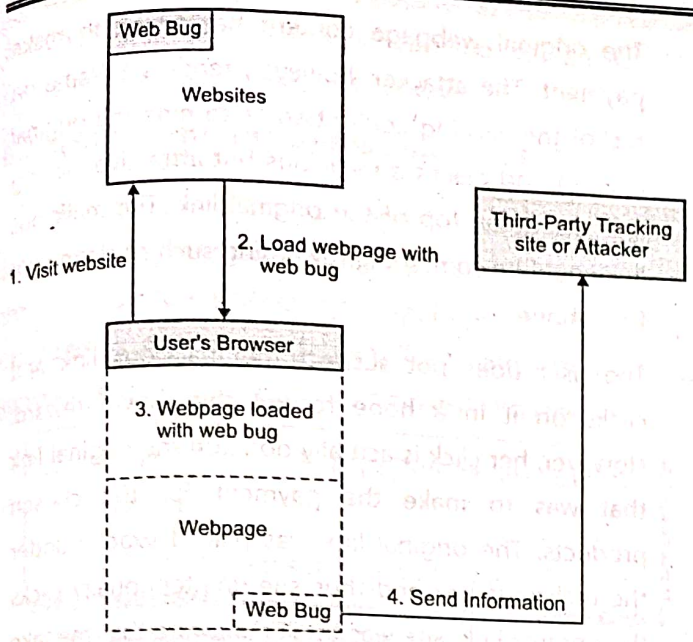


Fig. 6.7.3

1. The user, without any knowledge of web bug's presence on the website, visits the website.
2. The browser follows the normal way of loading the webpage when it receives the response from the webserver.
3. It loads the page content as well as the web bug. Usually the web bug loads its content from a different source (tracking server) than the page source (webserver) very much like how third-party trackers work.

   For example,

```
<img
src="http://ad.doubleclick.net/ad/pixel.quicken/NEW"
width=1 height=1 border=0>
```

   The browser cannot make out any difference between the legitimate content and the web bug content. The browser, as per its design, loads the content present on the webpage. The webpage content can come from various sources and the browser does not restrict which sources should be allowed and which not (at least not by default).
4. Once the web bug hits its source (tracking server), the tracking server can record all the typical information that is passed on to any webserver for serving the requests such as
   - The page the web bug came from
   - IP Address from which the web bug content request is coming
   - Browser information
   - OS information
   - Other fingerprinting information that you learnt earlier

   The tracking server could potentially also execute scripts that can collect additional information as desired.

(B) How Web Bugs Work on Emails?

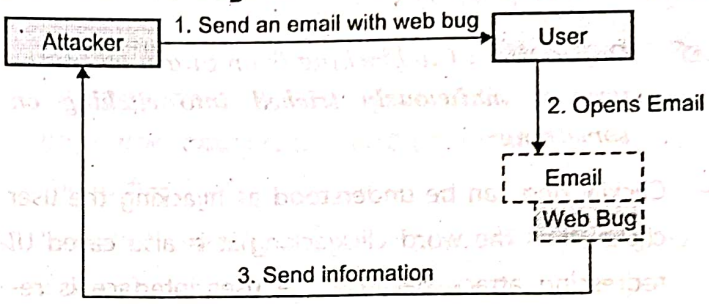


Fig. 6.7.4

Web bugs work very similar on emails as well.

1. An attacker or a company sends a html email containing the web bug. The email could contain promotional information, phishing information or could be just look like to be sent erroneously. The objective is to check whether the user opens such emails and if yes then collect the information that could be used to create further exploits.
2. The user may open such an email that contains the web bug. The web bug could contain an image that gets its content from the tracking server as you learnt earlier.
3. The web bug can then submit the information that a normal web request would send to the webserver. Additionally, it could convey information such as the time, the device and the software using which the email was viewed. The user may delete the email later but once it was opened, the information is already sent to the attacker. The user may forward the email to her contacts. It could then collect the information from various users as well.

6.7.2(A) Countermeasures against Web Bugs

Following are some of the recommended best practices to protect against web bugs.

1. Use browser extensions that can block loading of web bugs. These extensions are usually the same that block third-party trackers.
2. Do not open emails from untrusted sources.
3. Prefer plaintext emails over html emails. Additionally, you may configure your email client to not download images or load html content by default.

6.7.3 Clickjacking

✍ ***Definition :*** *Clickjacking is an attack where the user is maliciously tricked into clicking on something.*

- Clickjacking can be understood as hijacking the user clicks (thus the word clickjacking).It is also called UI redressing attack because the user interface is re-dressed (rendered) maliciously.
- The user is shown a webpage (or specifically a frame within the page) that hides the original page content (text, button, pictures, etc.) and invites the user to click on something very rewarding. The malicious page frame is layered on top of the original frame so as to hide the original frame and its content. When the user clicks on the malicious layer on the top, the click is actually transferred to the lower and the original layer. The user, thus, unknowingly clicks on something that she would have perhaps never clicked if the original content was visible.
- Let's take an example.

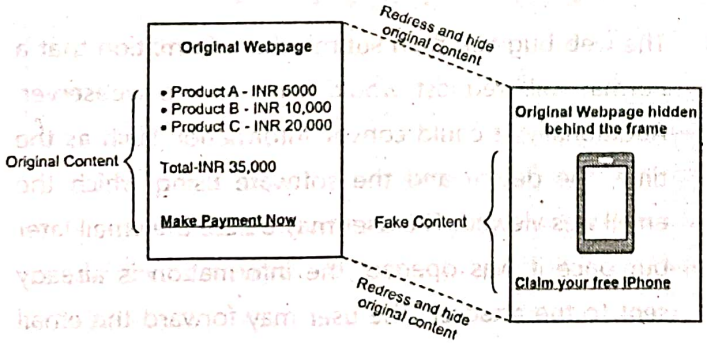


Fig. 6.7.5

- The original webpage content has a link to make payment. The attacker, however, renders a frame on top of the existing content so as to hide the original content and places a malicious but attractive looking link exactly on top of the original link. The malicious link could be something rewarding such as claiming a free phone.
- The user does not suspect the malicious link and clicks on it in a hope to get the shown reward. However, her click is actually done on the original link that was to make the payment for the chosen products. The original link was placed exactly under the malicious link and thus she unintentionally clicks the original link. She was totally unaware that the fake frame was layered on top of a different content.
- Typically, if a website is vulnerable to clickjacking attack, the attacker would be able to load its webpage inside a frame. For example, the following code would show the content of your page inside the frame if your website is vulnerable to clickjacking. Your webpage can be then hidden, and the attacker can layover her own content on top of your webpage.

```
<html>
<head>
<title>Clickjacking Test !!!</title>
</head>
<body>
<p>The website is vulnerable to clickjacking!</p>
<iframe src="http://www.example.com/webpage"
width="400" height="300">
</iframe>
</body>
</html>
```

There could be some minor variations of clickjacking attack.

Some of them are as following.

1. **Likejacking :** User is manipulated to click on the "like" button.
2. **Cursorjacking :** The mouse cursor's appearance and location are manipulated.

3. **Cookiejacking :** Instead of carrying out an action, cookies are stolen from the browser.

6.7.3(A) Countermeasures against Clickjacking

Clickjacking is a browser based vulnerability. There are client-side mechanisms such as browser extensions that can protect against clickjacking. However, server-side mechanisms are more effective. There are majorly two server-side techniques to protect against clickjacking.

1. Content Security Policy (CSP) frame-ancestors directive
2. X-Frame-Options Response Headers

1. Content Security Policy (CSP) Frame-Ancestors Directive

- The HTTP `Content-Security-Policy` response header allows website administrators to control resources that the browser is allowed to load for a given page. Policies specify server origins and script endpoints that the browser is allowed to load. This helps to guard against several attacks such as clickjacking and cross-site scripting (XSS).
- CSP uses several directives (instructions) to declare the policies. Its syntax is as following.

Content-Security-Policy: <list of policy-directives>

- Out of the several CSP directives, the `frame-ancestors` directive specifies valid parents that may embed a page using `<frame>`,`<iframe>`, `<object>`, `<embed>`, or`<applet>`.

Content-Security-Policy: frame-ancestors <source>;

Content-Security-Policy: frame-ancestors <source><source>;

The `source` could be the following.

1. **<host-source> :** This specifies the internet host by IP address or URL. For example, http://*.example.com would match all attempts to load from any subdomain of example.com using http:.
2. **<scheme-source> :** This specifies the type of URL scheme. It could be http, https, data, blob, filesystem or mediastream. So, for example, if you have specified that a particular content is to be served only over https, it won't be served on http URL scheme.
3. **'self' :** It refers to the origin from which the protected document is being served, including the same URL scheme, host name and port number. The protected document cannot be loaded from any other page. So, for example, if you navigate from page 1 to page 2 of the website, the page 2's origin must be same as page 1's origin.

   Origin is the combination of URL scheme (http, https), host name (or IP address) and port number. So, if page 1 was loaded from https://www.example.com on port 443, the page 2 of the website, for example, https:///www.example.com/protected.php must be on https, from example.com and on port 443 as well for successfully loading it.
4. **'none' :** It refers to no URL match. It does not allow loading or embedding the content.

   So, using the CSP `frame-ancestors` directive you can protect the content from server side to be loaded inside a frame. You can specify the `source` policy as per your site requirements. Following are the three examples.

Content-Security-Policy: frame-ancestors 'none'

Content-Security-Policy: frame-ancestors 'self'

Content-Security-Policy:frame-ancestors www.example.com

2. X-Frame-Options Response Headers

- CSP, as a protection mechanism, is fairly new and obsoletes the `X-Frame-Options` Response Headers. On earlier versions of browsers, `X-Frame-Options` Response Header was used to protect against loading the protected content in frames.
- The `X-Frame-Options` HTTP response header can be used to indicate whether a browser should be allowed to render a page in a `<frame>`, `<iframe>`, `<embed>`or`<object>`. You can use this to avoid clickjacking attacks, by ensuring that your website content cannot be embedded into other websites.

- Note here that the `X-Frame-Options`HTTP Response Headers will provide protection only if the user is using a browser that supports `X-Frame-Options`HTTP Response Headers.
- Like CSP, if a browser does not support `X-Frame-Options`HTTP Response Headers, the user would not get the protection. Hence, it is important for you as a user to choose the right browsers that support such protection mechanisms.
- `X-Frame-Options` supports three directives.

1. **deny :** The deny directive ensures that the webpage cannot be displayed in a frame on any website.
2. **sameorigin :** The sameorigin directive ensures that the webpage can only be displayed in a frame on the same origin (URL scheme, hostname and port number must be same) as the page itself. You now understand what is sameorigin from previous discussion.
3. **allow-from :** The allow-from directive ensures that the webpage can only be displayed in a frame on the specifically specified origin.

- You could use the `X-Frame-Options` like CSP with the directives as appropriate for your website.

X-Frame-Options: deny

X-Frame-Options: sameorigin

X-Frame-Options: allow-from https://example.com

6.7.4 Cross-Site Request Forgery (CSRF)

✍ ***Definition :*** *Cross-Site Request Forgery (CSRF) is an attack where the user unintentionally authorises malicious instructions to a web application.*

CSRF involves executing instructions (change state of the application or data) unlike just stealing the data. It is also known as one-click attack or session riding or XSRF. Unlike Cross-site Scripting (XSS) attack where the user is attacked by the trusted web application (through the malicious code injected by the attacker into the web application), CSRF involves attacking the application by exploiting the trust that the application has on the user (or more specifically the established and active user session).

6.7.4(A) How CSRF works?

Let's see an example of how this could work. Following is a simple block diagram of interactions that could be possibly taken for a CSRF attempt.

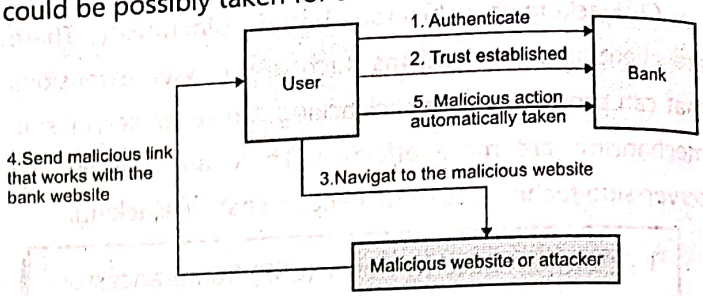


Fig. 6.7.6

1. The user successfully authenticates with the banking website say https://www.mybank.com.
2. Now, you understand that the website would place a cookie that would save the session state as the user continues her interaction with the banking website. The user is not required to login again until she logs out or the session expires. The session would remain valid and any protected webpage or the action request on the banking website would be served without asking the user to authenticate again.
3. Now, suppose that the user is also side-by-side browsing and goes to a malicious or infected website. The website, looking to exploit CSRF, has a pre-prepared link that could carry out some action (say bank transfer) if requested by the authenticated user. Assume that the link for such an action is https://www.mybank.com/transfer?to=987654;amount=20000. Without the previous successful authentication (as in step 1), if a user sends this link to the bank's website, it will be rejected. But, if the link is sent to the bank website after successful authentication, it would be accepted, and the desired action would be carried out.
4. The malicious website can use the malicious link on its webpage say within the <img> HTML tag as following.

```
<img
src="https://www.mybank.com/transfer?to=987654;amount=20000" width="0" height="0" border="0">
```

5. When the user's browser tries to load the malicious website, it hits the malicious link hoping to retrieve the image content as specified in the <img> tag. The browser has no clue of what the link is about and where it would be sent. The user is authenticated (as in step 1. The malicious link is sent to the bank website and the transaction is successfully carried out without user's intention or knowledge because the user's session is still active and valid.

6.7.4(B) Countermeasures Against CSRF

CSRF is a well-known vulnerability and there are quite a few mechanisms using which it can be addressed. Following are the major countermeasures for preventing CSRF.

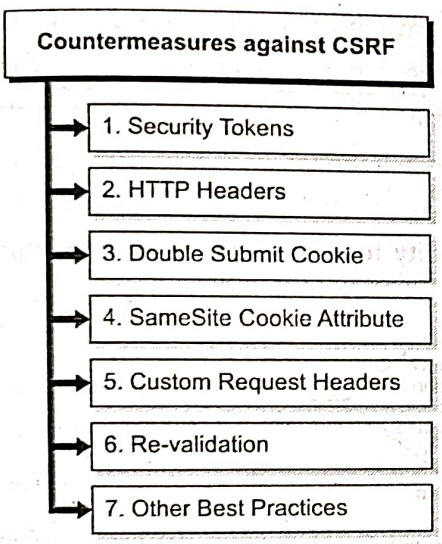


Fig. 6.7.7

1. **Security Tokens :** Token based mechanism to prevent CSRF is one of the most popular and effective mechanisms. As you understand that CSRF involves changing the state of the application or data. You can enforce that any state changing action is accompanied by a security token that validates that the action is indeed coming from the user and is not manipulated using CSRF.

- A CSRF security token is unique per user session. It is a large random value and can also be generated by a cryptographically secure random number generator. The security token can be added as a hidden field for forms or could be preferably passed via HTTP headers. Note here that the security token is not same as the cookie that is sent on every request automatically by the browser. It is more like a session ID that is hard to guess and is valid only as long as the session is valid and active.
- The website sets the CSRF security token once the session is established. It would then no more accept links such as
- `https://www.mybank.com/transfer?to=987654;amount=20000` and would require that each state changing request is accompanied by the security token established earlier. So, the link must be something like

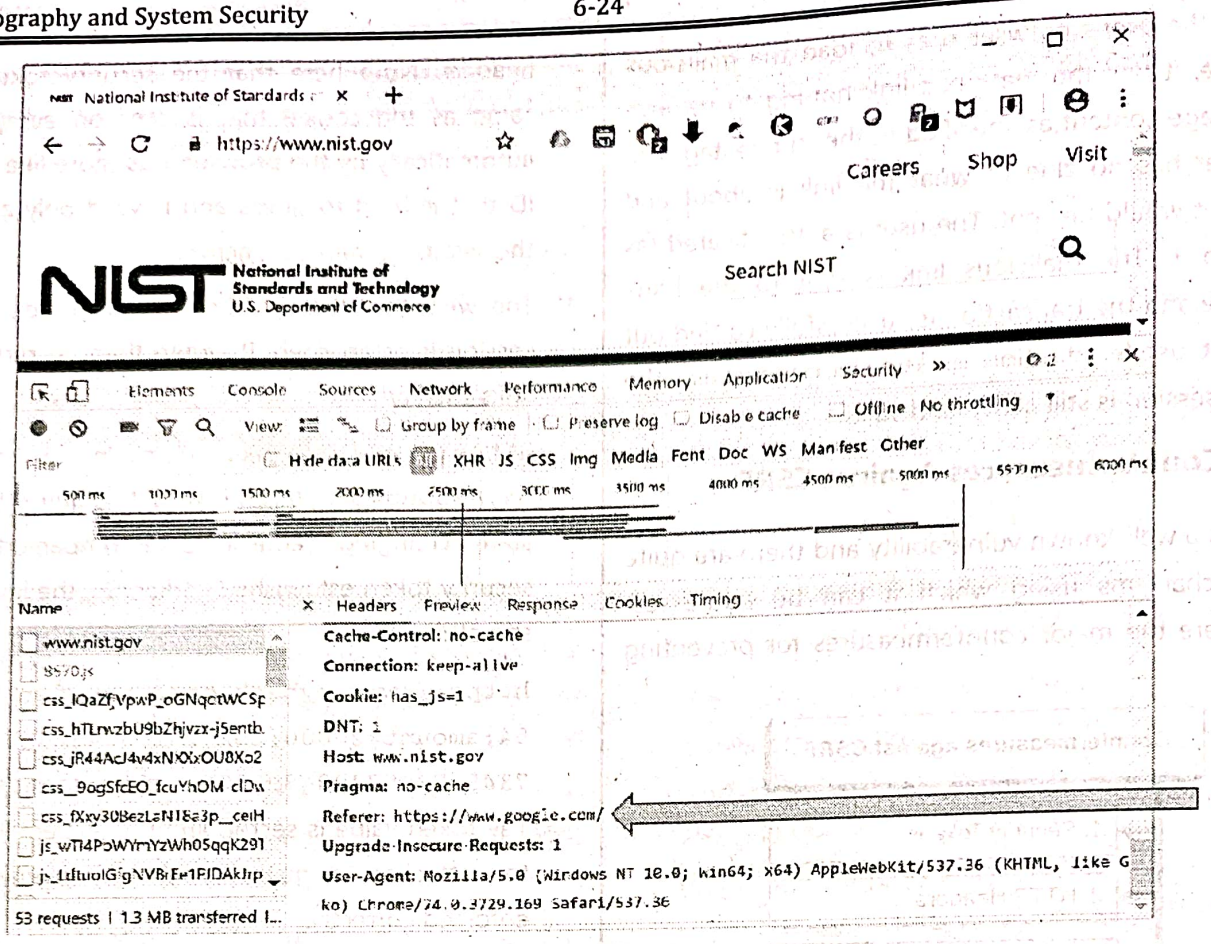
`https://www.mybank.com/transfer?to=987654;amount=20000;token=21374364273423541734517345749848924562` to be considered valid. The token value is secret, and it is extremely hard for the attacker to guess it when implemented and enforced correctly.

2. **HTTP Headers**

- You can use the HTTP headers to determine the source of request. If the source of request is not the site that is expected, the request should be rejected.
- The `Referer` header indicates the URL which initiated the request received by the webserver. It can distinguish a same-site request from a cross-site request because it contains the URL of the website making the request. A website can defend itself against cross-site request forgery attacks by checking whether the received request was issued by the site itself or it is being sent from a different website. It's typical syntax is as following.

Referer: https://example.com

- Look at the following example snapshot. I searched for NIST using Google search and clicked on the first link that appeared in the search results. Because I went to the NIST website from Google search, the `Referer` header is set to https://www.google.com/ indicating that I landed on the page from Google search.



3. **Double Submit Cookie :** In scenarios where storing the CSRF security token on the server-side is problematic, double submit cookie technique may be used.

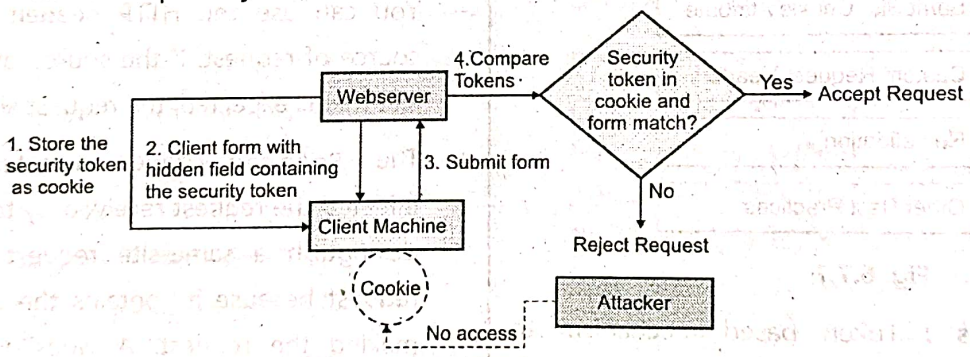


Fig. 6.7.8

– n the double submit cookie technique, the security token is stored as a cookie and is also submitted as part of the form page from the client as a hidden field. The server checks if the token value received from the cookie and the form's hidden field match. If they match, the request is accepted else it is rejected. The attacker cannot read or set the cookie value due to the same-origin policy and hence cannot manipulate the form to contain the valid security token and send it to the webserver.

4. **SameSite Cookie Attribute :** `SameSite` is a cookie attribute similar to `HTTPOnly`, `Secure` etc. This attribute helps in preventing the browser from sending cookies along with cross-site requests. Possible values for this attribute are `lax` or `strict`.

– Setting the value of the `SameSite` attribute to `strict` would prevent sending cookies in any cross-site references. If a cookie has this attribute, the browser will only send it if the request originated from the website that set the cookie. If the request originated from a different URL than the URL of the current location, then the cookies tagged with the `strict` attribute will not be sent.

- Setting the value of the `SameSite` attribute to `lax` would selectively allow sending cookies in cross-site references. The requests such as calls to load images or frames would be denied. But, the cookies will be sent when a user navigates to the URL from an external site, for example, by following a link.
- You could set the SameSite cookie attribute as following.

```
Set-Cookie: SESSIONID=1234; SameSite=Strict
Set-Cookie: SESSIONID=3456; SameSite=Lax
```

5. Custom Request Headers

- Custom HTTP headers can be used to prevent CSRF. These are the headers that the website itself decides to use for its own usage.
- The browser prevents a website from sending custom HTTPheaders to another website but allows it to send the custom HTTPheaders to itself using `XMLHttpRequest`.Awebsite must issue all the state changing requests using `XMLHttpRequest` and attach the custom header `X-Requested-By`.It can verify the presence of the custom header and its value and accept the request only if they are valid. If a request is not accompanied by the custom header or has an incorrect value set, then the website must reject the request.
- This approach has the double advantage of usually requiring no UI changes and not introducing any server side states. You can add your own custom header and its value as required.

6. Re-validation

- The techniques, that you learnt so far to prevent CSRF, did not require any user interaction. They were all programmatic methods to prevent CSRF. However, if you recall, CSRF becomes possible when another website takes advantage of the active trust already established between the user and the relying website. Hence, the CSRF requests can pass through without requiring any user interaction.

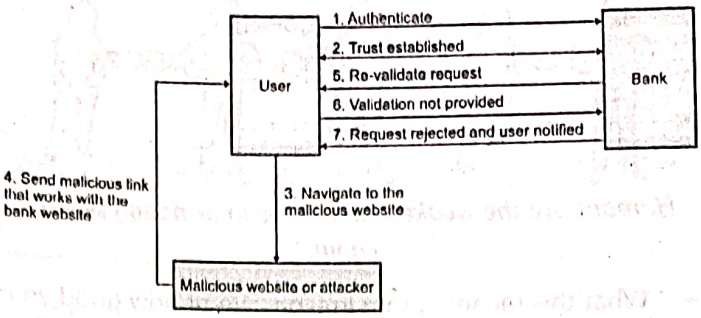


Fig. 6.7.9

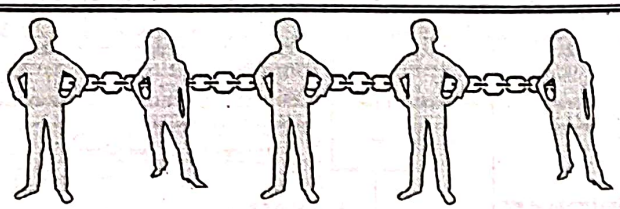
- Re-validation requires that any state changing transaction is re-confirmed from the user. This may require sending an OTP or re-entering the password. Banking websites typically implement this. An OTP is usually required to confirm a money transfer (or when adding a new payee) even if you are successfully logged in into the banking website. Re-validation can ensure that the state changing transactions are actually approved by the user.
- The block diagram presented here has the same steps that a typical CSRF has as you learnt earlier. Instead of the malicious link directly getting accepted by the bank, this time, the bank website requires a re-validation before processing the request. The user has an opportunity to deny the validation and hence reject the CSRF request.

7. Other Best Practices : On a personal front, it is recommended to follow the following guidelines for avoiding CSRF.

1. Logoff immediately once you have completed interacting with a website specially the sensitive ones such as banking.
2. Be selective in opening multiple tabs in a browser when carrying out sensitive transactions. It is better to open just one tab when carrying out the sensitive transaction and open others only when you have logged off after completing the transaction.
3. As usual, be selective in opening emails, browsing websites and clicking on links.

6.7.5 Social Engineering

- The information security domain experts strongly believe on a key point:



Humans are the weakest link in the information security chain.

- What this means is that irrespective of how good your technical controls (such as firewall, access control, password quality) are, people can potentially be tricked to bypass those technical countermeasures.
- So far, you have seen several technical ways in which an attacker might exploit a vulnerability to harm the victim. Social engineering is a non-technical way to harm.

✍ ***Definition** : Social engineering is the act of tricking or deceiving a person into giving confidential or sensitive information that could then be used against her or her organisation.*

- The attacker tries to be "friendly" with the victim and drives the conversation as per her pre-determined social engineering attack strategy. The mode of communication could be telephone, chat, email, SMS, face-to-face or anything else. The victim, without realizing the impact of sharing the sensitive information and without verifying the legitimacy of the conversation, joyfully shares the sensitive or the attacker-desired information with the attacker. Social engineers heavily use the human psychology to artfully craft an attack strategy to exploit human behaviour.
- Out of the several social engineering techniques, I would focus on phishing and its variants.

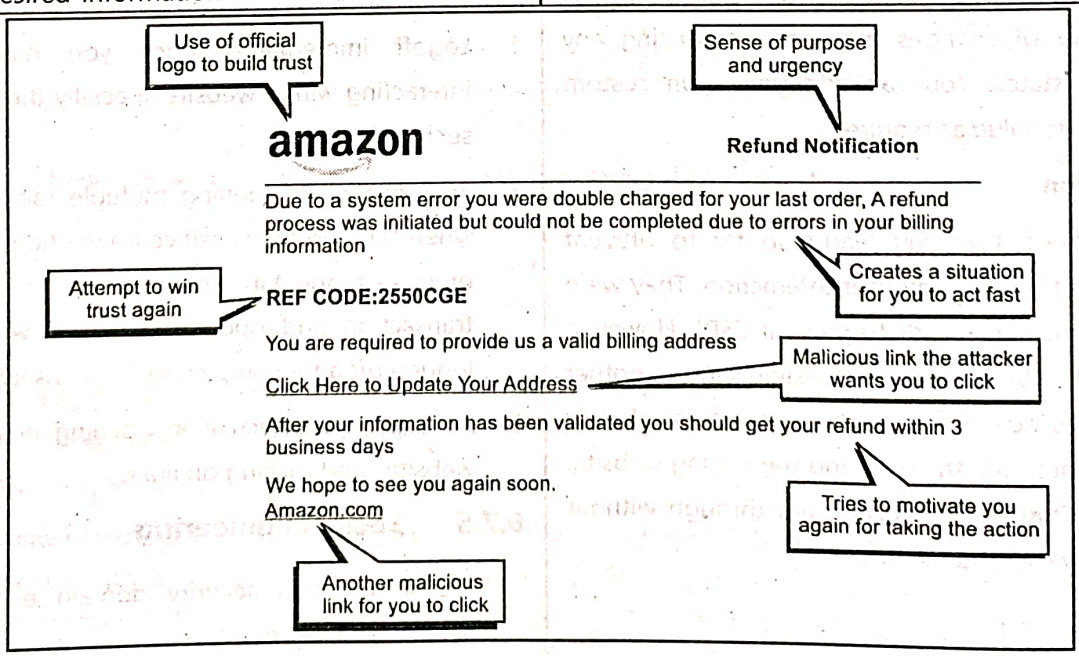
(a) Phishing

✍ ***Definition** : Phishing is a technique for attempting to acquire sensitive data through a fraudulent solicitation in email or on a website, in which the attacker pretends to be a legitimate business or reputable person.*

It tricks the individuals to disclose the sensitive information by claiming to be a trustworthy entity. It is the most common social engineering attack. It is usually carried out using email or text messages. Phishing messages create a sense of urgency, curiosity or fear requiring immediate action from the victim to stop any damage. It focuses on sending out high volumes of generalized emails with the expectation that at least a few people will be trapped. The victim's action could be

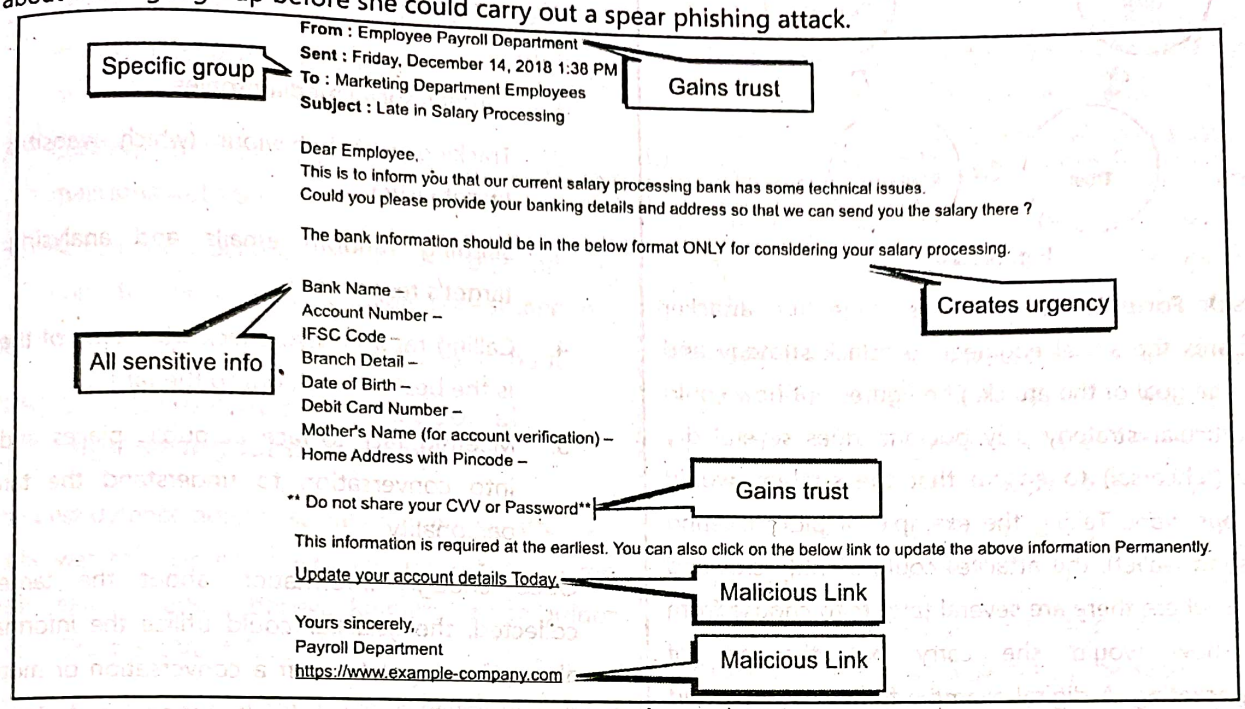
a. Clicking on malicious links
b. Opening malicious attachments or
c. Performing any other actions as the attacker desires

Here is an example of phishing email.



(b) Spear Phishing

Unlike phishing attacks, spear phishing attacks are carried out only on a selected group of few individuals. The email messages are highly customized and specifically targeted to only the selected group. For example, the group could be a division of an organisation. The attacker needs to do extensive research and gather enough information about the target group before she could carry out a spear phishing attack.



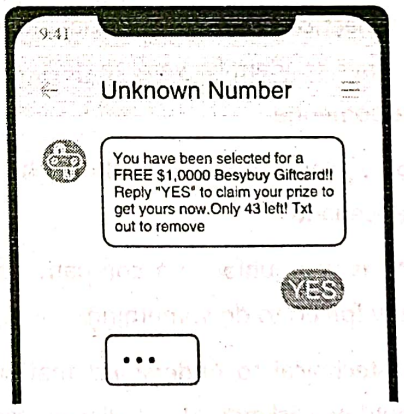
(c) Vishing

Vishing (or voice phishing) is a social engineering attack that uses a telephonic conversation for soliciting (getting) information. For example, you could receive a call with a recorded message that states that your credit card has been breached and to call a particular phone number immediately. When you call the given phone number, you are asked to verify your 16-digit card number, date of birth, phone number, home address, mother's name or any other sensitive information. You have provided a wealth of sensitive information that could be used to harm your financial status.

Note: Just in case, you are interested, I would suggest that you watch a short video about a live vishing attack and how trust could be easily established. Visit the short YouTube video URL https://tinyurl.com/hoo8wwq. This would open a YouTube video located at https://www.youtube.com/watch?v=lc7scxvKQOo.

(d) Smishing

Smishing (SMS-Phishing) uses SMS for tricking the targets to perform the desired actions.



6.7.5(A) Social Engineering – The Attack Cycle

- Following is the typical lifecycle of any social engineering attack including phishing.
- The typical social engineering attack is just like stealing someone's wallet. The phases are more or less the same. The social engineering attack cycle typically consists of 5 stages.

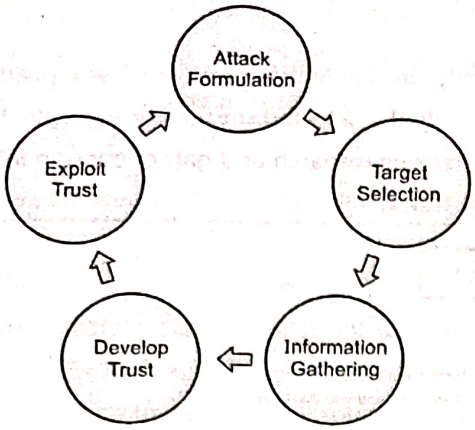


Fig. 6.7.10

1. **Attack Formulation :** At this stage, the attacker prepares the social engineering attack strategy and sets the goal of the attack. She figures out how could a particular strategy play out and does several dry runs (rehearsal) to ensure that the strategy would perhaps work. Taking the example of pickpocketing (stealing wallet), the attacker could identify crowded places where there are several targets to choose from and how would she carry out the act of pickpocketing. A digital example for this stage could be crafting an email to resemble Income-Tax department or say CEO of a company to extract sensitive information from the email recipients.
2. **Target Selection :** This is a crucial stage where the attacker tries to identify weak targets. A weak target could be someone
   1. Non-vigilant or uncaring about the situation or the scenario
   2. Who is very junior in a company and could be easily forced to do something
   3. Non-technical to understand that not all URLs should be clicked and not all websites are safe
   4. Who does not understand that not all information can be shared
   5. Who does not verify the identity or the conversation before providing information or performing actions
   6. Who easily believes everyone, and everything told or explained in a particular manner

   – The target is selected based on the criteria that best matches the attacker's strategy and goal.
3. **Information Gathering :** Once the target is selected, the attacker tries to collect more and more information about her. Information could be collected from
   1. Target's social media profiles
   2. Tracking web-behaviour (which websites the target visits)
   3. Sending random emails and analysing the target's response
   4. Calling randomly to check what time of the day is the best to reach out to the target
   5. Meeting face-to-face at public places and get into conversation to understand the target's personality

   Once enough information about the target is collected, the attacker could utilize the information about the target to craft a conversation or motivate the target to carry out an action on a particular website.
4. **Develop Trust :** At this stage, the attacker tries to build rapport with the target. The attacker makes the target believe that she is someone who could be believed and must get the target's attention and response. This trust could mean
   1. Trusting an incoming email
   2. Trusting a caller
   3. Trusting a scenario or a situation
   4. Trusting an unsafe website
   5. Trusting a SMS or chat
   6. Trusting a person to be someone (say a police officer)

   – The target at this stage sees no harm in sharing the information or performing actions as desired by the attacker.

5. **Exploit Trust :** Finally, once the target has some sort of trust on the attacker, this trust could be exploited to get information or perform the desired actions. The exploitation could mean

   1. Sharing sensitive, personal or confidential information such as credit card number or date of birth
   2. Clicking on URLs that take the target to the malicious websites
   3. Performing actions such as deleting an important file or sending an email to someone
   4. Transferring money into someone's account

6.7.5(B) Countermeasures against Social Engineering (and Phishing)

- The best defence against social engineering attacks is to be watchful. Do not trust everything that you see, hear, read or made to believe. Remember one golden rule – "Trust but Verify".
- Following are some of the ways in which you could avoid falling into the social engineering trap.

1. As an individual

(a) Do not open email attachments from someone you don't know.

(b) Do not provide sensitive information such as credit card number, bank account number, date of birth, mother's name, passwords, PIN, Aadhar number, etc. to anyone.

(c) Do not click on every URL you get in any email or ads on the websites you visit howsoever tempting it might look.

(d) Whenever in doubt always information your nearest elder, supervisor, police officer, organisation spokesperson or anyone else that could help regarding the matter

(e) Do not use weak and same passwords on all accounts. Wherever possible, use multi-factor authentication (requiring, for example, an OTP along with password for login). Do you use only password to login to your Facebook account? If so, please enable multi-factor authentication on it (and other accounts you have) today.

2. As an organisation

(a) Provide training to all employees and staffs on social engineering and general cybersecurity best practices periodically (minimally once a year) irrespective of the role or the position in the organisation. Remember that the people are the weakest link in security. Even a peon of an organisation may provide sensitive information to the attacker.

(b) Have a corporate cybersecurity program to ensure that the information systems are adequately protected and make it hard for the sensitive information to leave the organisation.

(c) Setup a team internally to whom the organisation's employees and staffs can reach out to or report to if they have any questions, tips, information and suspicion to share regarding cybersecurity or social engineering matters.

6.7.6 Pharming

✍ ***Definition :*** *Pharming is an attack technique in which the attacker corrupts an infrastructure service such as DNS (Domain Name Service) causing the user to be misdirected to a forged website.*

The user could then give out sensitive information assuming that the website is real. The user might also download infected software, malware or be exploited in other fraudulent ways.

How Pharming Attack Works?

Following is a simplistic view of pharming attack.

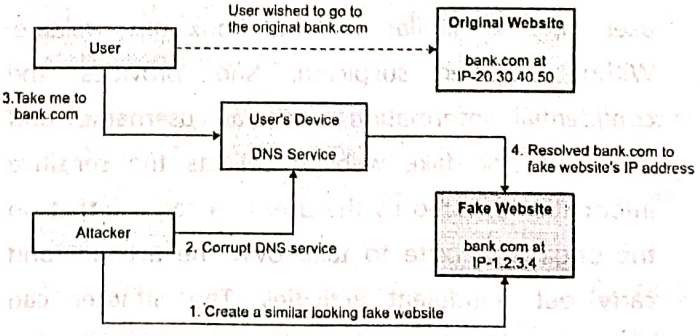


Fig. 6.7.11

1. First the attacker creates a similar looking fake website. The attacker need not create the entire website but just the pages that matter. For example, the attacker could just create the home page of bank.com where the user can provide login information to access her account.
2. The attacker then corrupts the DNS service on the user's device. This is usually done by adding the record for the fake website in the /etc/hosts file. The /etc/hosts file is given precedence over the DNS server configured on the user's device when the browser attempts to resolve the hostname to IP address. An entry in the /etc/hosts file could look like the following.

1.2.3.4 bank.com

On Windows® system, the file is usually located at C:\Windows\System32\drivers\etc.

Note : The attacker might carry out other DNS attacks such as DNS cache poisoning for corrupting the DNS service for the user. Manipulating the /etc/hostsfile is just one of the examples of corrupting the DNS records and carrying out the pharming attack.

3. When the user wants to go to bank.com website, she does not know which IP address to go to. DNS service is used for translating hostname to IP address so that the traffic can be reached to the appropriate bank.com's webserver. She just types bank.com on her browser.
4. Browser relies on the DNS for translating the hostname entered by the user to its corresponding IP address. The entry in the /etc/hosts file translates the bank.com website to the fake IP address 1.2.3.4. The user sees a similar looking bank.com website. Without further suspicion, she provides the confidential information such as username and password. The fake website collects the sensitive information entered by the user and can use that on the original website to take over the account and carry out fraudulent activities. The attacker can redirect the user to the original website after collecting the information to avoid suspicion.

Countermeasures against Pharming

Following are some of the ways to protect yourself from pharming attack.

1. Use secure DNS servers : Ensure that the DNS servers that you use are hard to infect or corrupt. Following are some of the public DNS servers, that are kept secure by top companies and communities, that you could use.

Company Name	DNS Server
OpenDNS	208.67.222.222 and 208.67.220.220
Cloudflare	1.1.1.1 and 1.0.0.1
Google	8.8.8.8 and 8.8.4.4
Norton ConnectSafe	199.85.126.10 and 199.85.127.10
Comodo Secure DNS	8.26.56.26 and 8.20.247.20
Quad9	9.9.9.9 and 149.112.112.112
Verisign DNS	64.6.64.6 and 64.6.65.6

These DNS servers provide several security and privacy features.

2. Protect your system from malware. Usually, malware can modify your DNS configuration and lead you to pharming attack.
3. Pharming is closely related to Phishing in terms of carrying out the exploit. Follow the recommended best practices for phishing.

6.7.7 DNS Attacks

- DNS (Domain Name System) is the backbone of the network. It translates the hostname into IP address so that the appropriate machine can be reached. With such a widespread impact of DNS on the network, it is naturally a common attack target.
- Following are some of the major attacks against DNS.

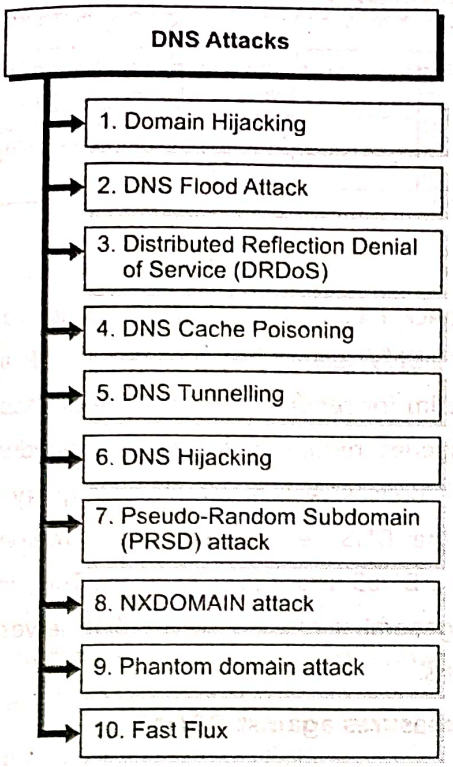


Fig. 6.7.12 : DNS attacks

1. Domain Hijacking

✍ ***Definition :*** *Domain hijacking is an attempt to maliciously own a particular registered domain.*

- For example, suppose you own abc.com and your entire business runs under that particular domain name branding. In domain hijacking attack, the attacker becomes the owner of abc.com. This could mean not only the loss of business for you but also could have a tremendous impact on your business goodwill, trust and reputation. In the era of web presence, domain name is extremely important and hijacking it could mean stealing or taking ownership of your business.
- The attacker can use the hijacked domain to carry out fraudulent activities or bring up a fake website and collect the sensitive details of the users as they try to login on the fake website.
- Domain hijacking is not a direct attack on the DNS itself but an attack on the DNS registrar system that impacts DNS activities. The DNS registrar is responsible for registering and operating the domain on behalf of its owner. Domain hijacking can also be carried out by taking over the account of the domain owner through social engineering attack or by carrying out any other regular account takeover attacks such as exploiting weak passwords.

Countermeasures against Domain Hijacking

Following are some of the countermeasures against Domain Hijacking.

1. The Internet Corporation for Assigned Names and Numbers (ICANN) is a non-profit organisation responsible for coordinating the maintenance and procedures of IP addresses and domains. It imposes a 60-day waiting period between a change in the registration information and a domain transfer to another registrar. This is intended to make domain hijacking more difficult, since a transferred domain is much more difficult to reclaim, and it is more likely that the original registrant will discover the change in that period and alert the registrar.
2. Extensible Provisioning Protocol is used for many Top-Level-Domain (TLD) registries and uses an authorisation code issued exclusively to the domain owner as a security measure to prevent unauthorised domain transfers.
3. As a domain owner you should protect your account using strong password and multi-factor authentication wherever possible. You should be careful with respect to social engineering and other account takeover attacks. Also, monitor your DNS registration for any fraudulent activities and let the registrar know as soon as possible.
4. Enable Domain Locking. Domain Locking is a mechanism using which domain name transfers can be locked against unauthorised changes.

2. DNS Flood Attack

✍ ***Definition :*** *DNS Flood Attack is a regular DDoS (Distributed Denial of Service) attack to make DNS servers non-responsive to legitimate requests.*

The attacker sends so many requests to the DNS server that its resources are exhausted and hence

would be unavailable to serve the legitimate requests coming from the real user. If DNS service is down, it could mean that the user is not able to browse the internet just by typing the hostname in the browser.

Countermeasures against DNS Flood Attack

As a countermeasure, you should follow the regular DDoS protection techniques.

1. **Reduce attack surface area :** Reducing the attack surface area means reducing the number of points exposed for attacks. These could be network ports, number of services running, open networks, unrestricted administrative access, unpatched OS or applications or anything else that could be exploited.
2. **Plan for scale :** DDoS attacks target the limited resources. If you have resources spread over a large scale (or say cloud computing), the DDoS attack impact could be very minimal and could be absorbed by the large resource pool without impacting availability.
3. **Know what is a normal and an abnormal traffic :** Knowing your network traffic patterns would let you plan for action if you see any change in the pattern that signals that you are under attack. You can then appropriately take actions to remedy any impact from DDoS.
4. **Deploy firewalls :** DDoS attacks are mostly network based. Having stringent (strong) firewall rules, such that only appropriate traffic is allowed, is a great way to ensure that the resources are not wasted in serving the DDoS traffic. Only the traffic that is legitimate is allowed on the network. Rest all the traffic is dropped without impacting the resources.

3. Distributed Reflection Denial of Service (DRDoS)

✍ ***Definition :*** *Distributed Reflection Denial of Service (DRDoS) is an attack in which DNS servers are used to send a large volume of DNS response data to the victim to cause a DDoS attack.*

Let's see how it works.

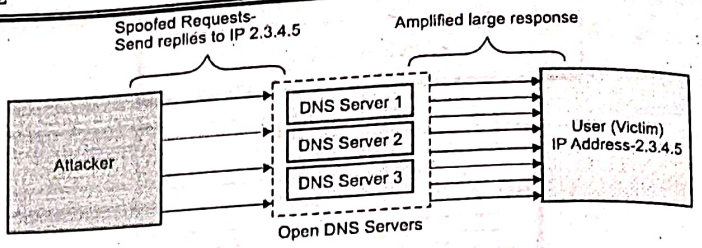


Fig. 6.7.13

The attacker sends multiple DNS queries to the open (public) DNS servers but provides the IP address of the victim for sending the response of those queries. The queries require much smaller bandwidth than their responses. As a result, multiple query responses from the DNS servers overwhelm the victim and create a DDoS attack for the victim. This victim could be a general user or a server that serves the user requests.

Countermeasures against DRDoS

Following are some of the countermeasures against DRDoS.

1. As a user, use a firewall to drop multiple DNS responses.
2. As a public DNS server operator, implement mechanisms to blacklist IPs that repeatedly follow an attack pattern. You could also use DNS monitoring solutions to send alerts when such attacks are attempted.
3. Use other DDoS protection mechanisms as outlined previously.

4. DNS Cache Poisoning

✍ ***Definition :*** *DNS Cache Poisoning is an attack in which the attacker can trick the DNS server to cache false information (hostname to IP address mapping).*

- It is also called DNS spoofing.
- Typically, the DNS servers cache the hostname to IP address mapping information so that they do not have to check the database or DNS registries frequently and optimise the response time. Now, if the attacker can manipulate the cache such that the hostname resolves to an incorrect IP, the DNS server

would continue to provide the incorrect IP address as a response to the hostname resolution until the TTL (time to live) value for the cache expires.

- Let's take an example.

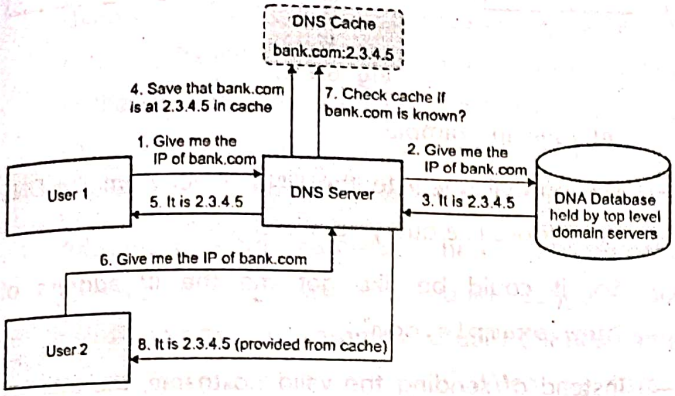


Fig. 6.7.14

- Under normal operations, following is how DNS cache works at a high-level.

1. For the first time when the DNS server is queried for resolving a hostname, it is unaware of the IP address for that hostname.
2. It follows the DNS resolution process and goes till the top level domain name servers to get the IP address of the queried hostname.
3. The top level domain name server provides the IP address for the queried hostname.
4. The DNS server then saves that IP address to hostname mapping in its cache to avoid following the DNS resolution process again and optimise the response time.
5. It provides the response to the first user by resolving the hostname to the received IP address.
6. Now, assume that it receives the query to resolve the same hostname again.
7. It checks its cache to see if the hostname to IP address mapping is already available.
8. If the information is available, then the response is provided as saved in its cache. If the information is not available, then it has to follow the DNS resolution process again.

- Now, assume that the attacker can manipulate or block the response from the top level domain servers ordirectly manipulate the cache information somehow.

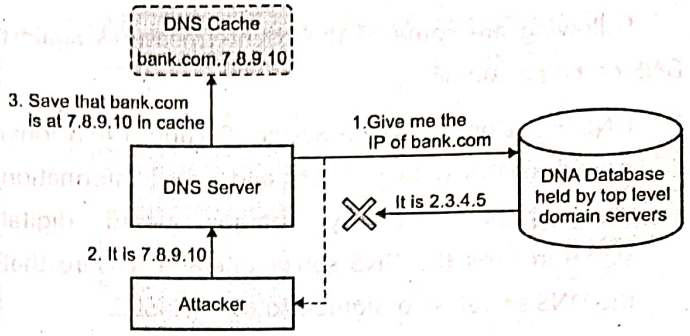


Fig. 6.7.15

- When the user queries the DNS server, she is provided a response from the poisoned cache and she is misdirected to the fake and malicious website.

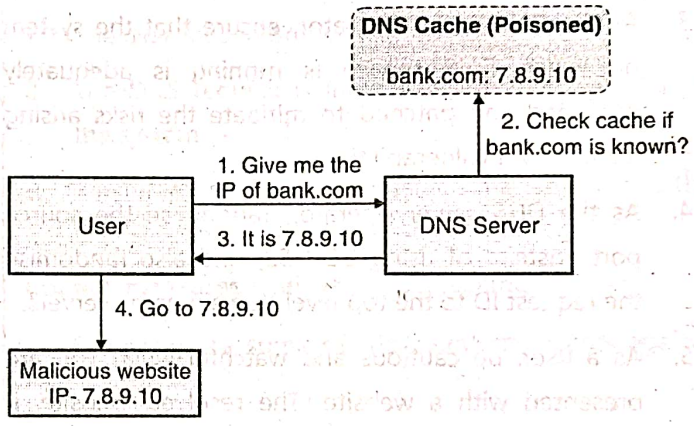


Fig. 6.7.16

- The attacker can poison the cache by faking herself as a top level domain server or by exploiting system vulnerabilities. The DNS protocol uses UDP and hence there is no acknowledgment for responses unlike TCP.
- However, it is not straight forward to poison the cache. The attacker would have to time the attack really well before the reply the from the legitimate top level domain server comes. Attacker would also need to know
  - Which hostnames are cached already, and which are not
  - Which top level domain server will the DNS server go
  - The request ID number and the port

– However, if there are known vulnerabilities on the DNS server itself, the cached information can directly be manipulated.

Countermeasures against DNS Cache Poisoning

Following are some of the countermeasures against DNS cache poisoning.

1. DNSSEC (Domain Name System Security Extensions) provides DNS data integrity and origin information using public key cryptography based digital signatures. As the DNS server operator, ensure that the DNS server is configured to use DNSSEC.
2. As the DNS server operator, use short TTL (time to live) values. DNS cache remains valid only until TTL expires. Using small TTL value would overwrite the DNS cache data and clear the fake entries.
3. As the DNS server operator, ensure that the system on which DNS service is running is adequately hardened and patched to mitigate the risks arising from known vulnerabilities.
4. As the DNS server operator, randomise the source port (instead of using UDP 53) and also randomise the request ID to the top level domain name servers.
5. As a user, be cautious and watchful when you are presented with a website. The resolved website, if fake, would mostly not have a valid TLS certificate. Do not ignore the browser warnings.

5. DNS Tunnelling

✍ ***Definition :*** *DNS Tunnelling is a mechanism to hide other communication protocols and data in DNS queries and responses.*

– DNS queries and responses are normally not suspected to carry data and are generally allowed through firewall. The attacker can use the DNS query and response mechanisms to hide and send commands and data to a remote network by bypassing the firewall and other network access controls. It is often used for data exfiltration (sending data in an authorised manner) or to bypass authentication for paid Wi-Fi.

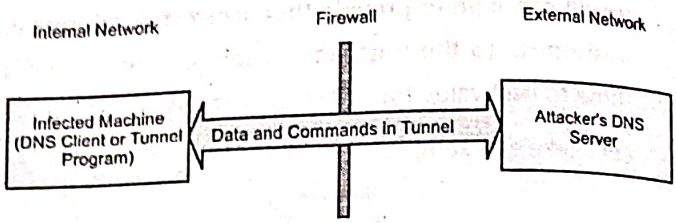


Fig. 6.7.17

– Let's see an example.

– The normal query to the DNS server from the DNS client looks like querying a hostname.

So, it could be like get me the IP address of "`new.example.com`".

– Instead of sending the valid hostname, the attacker can send something like get me the IP address of "`Y2FyZD0yMzQ1Njc4OTtjdnY9MTIzO2V4cGlyeT0xMTIzO25hbWU9Sm9l.example.com`".

– This looks to be a valid request and the firewall allows it.

But in reality "`Y2FyZD0yMzQ1Njc4OTtjdnY9MTIzO2V4cGlyeT0xMTIzO25hbWU9Sm9l`" is a base64 encoded string for "card=23456789;cvv=123;expiry=1123;name=Joe". The attacker's DNS server, which is aware of such queries coming from the infected machine, can then decode such strings and collect all the data.

Countermeasures against DNS Tunnelling

DNS tunnelling looks like normal DNS traffic. Traffic and pattern analysis are required to detect it. Following are some of the ways DNS tunnelling can be detected and protected from.

1. **Frequency of DNS requests :** As per the DNS protocol specification, the size of a DNS UDP message should be 512 bytes or less. Now, if an attacker is looking to transfer a huge volume of data, she would require making multiple DNS requests and sending a chunk of data with each request. So, in your environment, if you normally see 300 DNS requests per day and all of a sudden start to see like

1,000 requests per day, it could be suspected that the DNS tunnelling attack is ongoing.

2. **Length of DNS requests :** Usually, the hostnames that are queried for IP address are under 30 characters. When the attacker uses encoding to send data out, the length of hostname could really be long. In the example I gave earlier, the hostname has 68 characters. You can look for such long name patterns in DNS queries and detect DNS tunnelling.

3. **Patterns of DNS requests :** There could be other patterns of requests that could suggest DNS tunnelling attack. For example, usually the hostnames do not have a lot of numbers. When the data is encoded and sent as part of the query, it would have quite a few numbers in the hostname.

4. **Target DNS Server :** You could also check if the requests are going to the right DNS server or if the DNS server configuration is changed. Use of an unapproved DNS server could indicate malicious DNS activities and DNS tunnelling could be one of them.

5. **Protect your systems :** DNS tunnelling requires access to the system and installing of tunnelling software and programs. You should continuously monitor your systems to identify any unapproved software or programs running on them. Additionally, follow system security best practices such as hardening and patching to keep your system secure.

6. **Use monitoring solutions :** If your infrastructure environment is large, it could be humanly difficult to keep pace with monitoring your entire infrastructure. Use of monitoring solutions such as IDS / IPS / SIEM / Log management, etc. could help you identify DNS tunnelling attack patterns and alert if any attacks are suspected.

7. **DNS Hijacking**

✍ ***Definition :*** *DNS Hijacking is an attack in which the DNS requests are intercepted, and malicious DNS resolution is provided.*

- This is also called DNS redirection attack.
- Note here that DNS hijacking is different from DNS cache poisoning. While DNS cache poisoning is done on the DNS server machine, DNS hijacking is done locally on the user's machine. DNS hijacking can be done by the malware on the system or your Internet Service Provider (ISP) if it wants to redirect you to its partner websites instead of the regular websites.

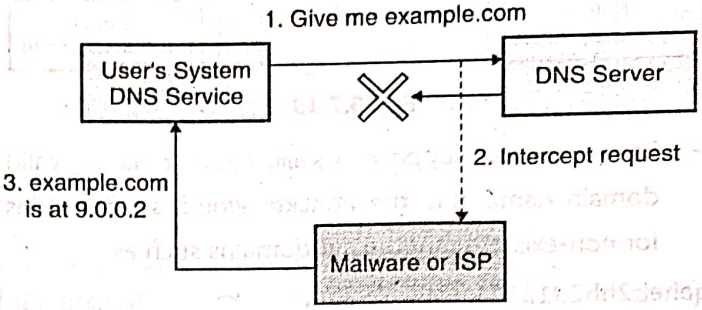


Fig. 6.7.18

1. The user tries to browse a website.
2. The DNS request is intercepted by the malware on the system or the ISP.
3. The intercepted request is served maliciously, and the user sees the malicious website.

Countermeasures against DNS Hijacking

Following are some of the countermeasures against DNS hijacking.

1. Keep your systems secure and free from malware.
2. Use public and open DNS servers maintained by reputed parties.
3. Use browser extensions that avoid DNS redirection.

8. **Pseudo-Random Subdomain (PRSD) Attack**

✍ ***Definition :*** *Pseudo-Random Subdomain (PRSD) Attack is a DoS attack where multiple non-existent subdomain queries are submitted by the attacker.*

- It is also called "Slow Drip" or "Water Torture Attack".
- The attacker uses bots or many infected systems to send multiple queries for non-existing subdomains such that to exhaust all the DNS server resources.

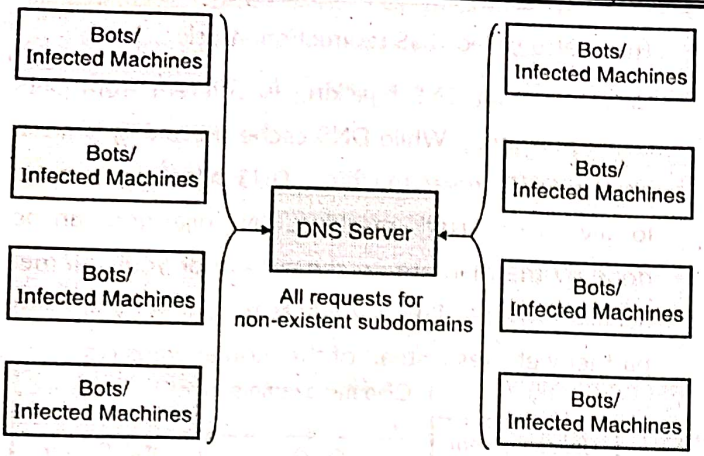


Fig. 6.7.19

– For example, suppose `example.com` is a valid domain name. But, the attacker would send queries for non-existent random subdomains such as

qeheb2hb2312321.example.com
vyhy74733282.example.com
fhu3rh3e823njgrj.example.com
pjkowey23774.example.com

– Because, example.com is a valid domain name, the DNS server would try resolving the subdomain name and would send queries to top level domain servers hosting the records for example.com. These non-existent subdomain queries, when sent repeatedly in a short period of time and in large volume, can exhaust the DNS server's resources (as well as top level domain server resources) and carry out a DoS attack.

Countermeasures against Pseudo-Random Subdomain (PRSD) Attack

Following are some of the countermeasures against Pseudo-Random Subdomain (PRSD) Attack.

1. You should have enough capacity and resources on the DNS server to withstand such attacks over a period of time.
2. Use protective solutions, such as firewalls, that can look at the query patterns and can drop traffic before hitting the DNS servers. These solutions can also blacklist the IPs submitting such malicious requests.

9. NXDOMAIN Attack

✍ ***Definition :*** *NXDOMAIN Attack is a DoS attack where the attacker sends multiple queries for random non-existent domain names.*

– NX is a short form for Non-eXistent. In NXDOMAIN attack, the attacker sends multiple queries for non-exiting domain names to exhaust the DNS server resources. Unlike the Pseudo-Random Subdomain (PRSD) Attack where the DNS server hosting the existing domain is targeted for DoS, the NXDOMAIN attack targets the DNS server in general.

– When the queried non-existing domain name is not found, the DNS server makes an entry in its cache that the queried domain name does not exist. Overtime, the cache can become full of such non-existing domain name entries and for each valid domain name query, the DNS server would have to carry out the entire domain name resolution process and cannot optimise its responses as before. This could slow down the response time of the DNS server and would consume its resources for resolution process for each query and can even mean DoS for legitimate requests.

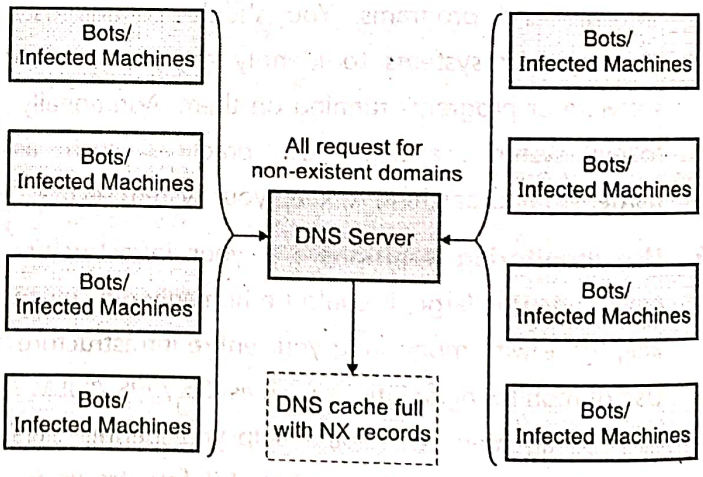


Fig. 6.7.20

Countermeasures against NXDOMAIN Attack

– Following are some of the countermeasures against NXDOMAIN Attack.

1. You should have enough capacity and resources on the DNS server to withstand such attacks over a period of time.

2. Use protective solutions, such as firewalls, that can look at the query patterns and can drop traffic before hitting the DNS servers. These solutions can also blacklist the IPs submitting such malicious requests.

10. Phantom Domain Attack

✍ ***Definition :*** *Phantom Domain Attack is a DoS attack where the attacker sends multiple queries for domains that are likely to respond slowly or never.*

- The dictionary meaning of phantom is "something existing in appearance only". Phantom domains are setup by the attacker purposely to respond very slowly or never. The DNS server is then hit by a flood of such requests and ultimately gets victimised for DoS.

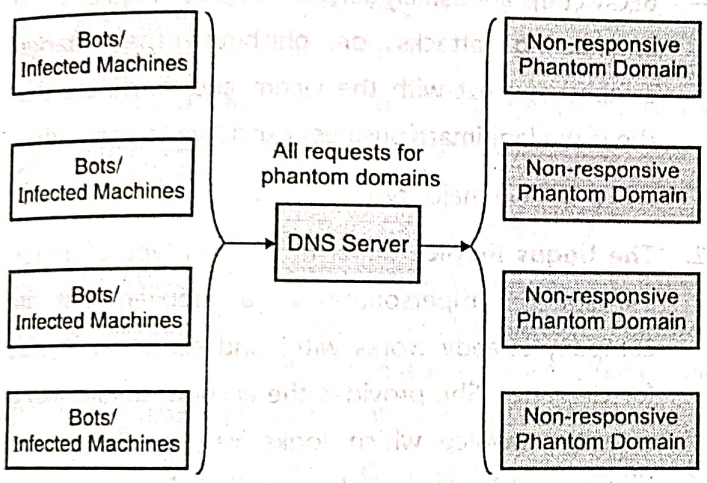


Fig. 6.7.21

Countermeasures against Phantom Domain Attack

- Following are some of the countermeasures against Phantom Domain Attack.

1. You should have enough capacity and resources on the DNS server to withstand such attacks over a period of time.
2. Use protective solutions, such as firewalls, that can look at the query patterns and can drop traffic before hitting the DNS servers. These solutions can also blacklist the IPs submitting such malicious requests.

11. Fast Flux

✍ ***Definition :*** *Fast Flux is a server hiding technique in which the target IP address for a domain name is changed rapidly.*

As you understand, the DNS server translates a hostname into its corresponding IP address based on the DNS records. Fast flux is a technique where the DNS records are changed rapidly (say under 5 minutes) such that to associate the domain name with a different IP address each time. This is done to avoid getting detected and getting the IPs blacklisted by the security solutions such as firewalls. The fast flux network is then used for malware distribution, phishing or carrying out other cybercrimes and malicious activities.

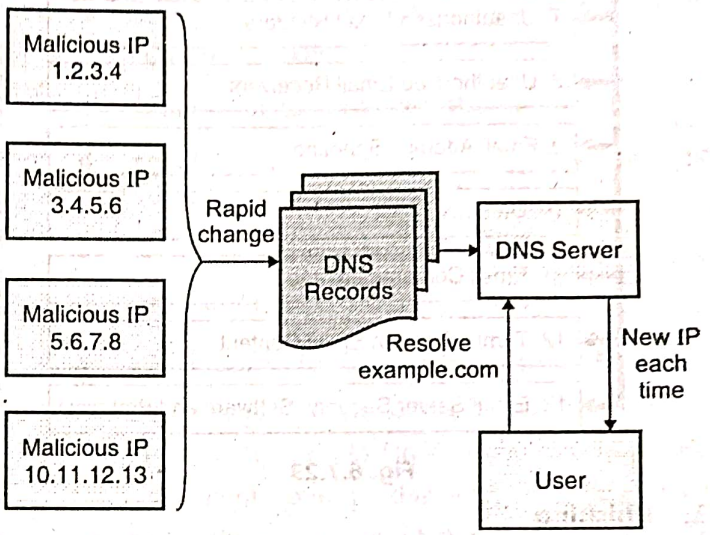


Fig. 6.7.22

Countermeasures against Fast Flux

- Following are some of the countermeasures against Fast Flux.

1. Fast flux technique requires the domain registrar to collaborate with the attacker to carry out changes to the DNS records such rapidly. As a domain operator, you should choose a reputed DNS registrars for hosting your website.
2. As a user, choose a reputed DNS server so that you are not served malicious pages intentionally.

6.7.8 Email Attacks

Emails have been the most common way of reaching out to the end users and victimise them. There are various forms of email based attacks. You have already seen quite a few of them but I will list them here for a sense of completion.

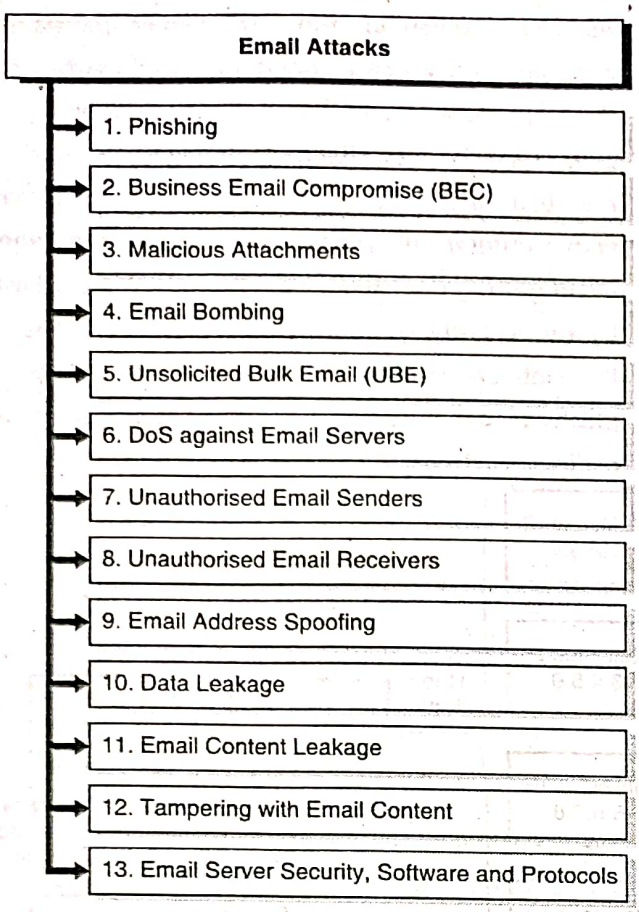


Fig. 6.7.23

1. Phishing

Phishing is perhaps the number one email based attack that has pretty high success rate despite various security campaigns run by various organisations. It is explained in detail in the "Web Browser Attacks" section previously in this chapter. Please refer to that section.

2. Business Email Compromise (BEC)

✍ ***Definition :*** *Business Email Compromise (BEC) is a type of spear phishing attack where the attacker impersonates a company's executive and tries to get a customer, an employee, or a vendor to transfer funds or to provide sensitive information.*

- It is also known as CEO Fraud. The person receiving the email trusts the authority and tone of the email and carries out the directed action. It might sound senseless, but it does really happen.
- As per the US FBI report dated 14 June 2016, BEC scam amounted to 3.1 Billion Dollar!

https://www.ic3.gov/media/2016/160614.aspx

STATISTICAL DATA

The BEC scam continues to grow, evolve, and target businesses of all sizes. Since January 2015, there has been a 1,300% increase in identified exposed losses[1]. The scam has been reported by victims in all 50 states and in 100 countries. Reports indicate that fraudulent transfers have been sent to 79 countries with the majority going to Asian banks located within China and Hong Kong.

The following BEC statistics were reported to the IC3 and are derived from multiple sources to include IC3 victim complaints and complaints filed with international law enforcement agencies and financial institutions:

Domestic and International victims:	22,143
Combined exposed dollar loss:	$3,086,250,090

The following BEC statistics were reported in victim complaints to the IC3 from October 2013 to May 2016:

Domestic and International victims:	15,668
Combined exposed dollar loss:	$1,053,849,635
• Total U.S. victims:	14,032
• Total U.S. exposed dollar loss:	$960,708,616
• Total non-U.S. victims:	1,636
• Total non-U.S. exposed dollar loss:	$93,141,019

- BEC victims are usually selected through regular social engineering attacks or phishing. The attacker establishes trust with the victim and convinces that she is the legitimate business executive to deal with.
- There are five major types of BEC.

1. **The Bogus Invoice Scheme :** In this type of attack, the attacker impersonates as a supplier (that the company already works with) and sends an invoice for clearance. She provides the account details along with the invoice which looks very similar to the invoice sent by the regular supplier of the company. The company's employee receiving the invoice and account information might consider it to be legitimate as in daily course of business and might make payment to the provided account without suspecting anything.
2. **CEO Fraud :** The attacker impersonates as the company's CEO or any other senior executive and sends an email to the finance department to transfer the money to a certain account. The email might require to urgently send some sensitive information instead of money transfer as well.
3. **Account Compromise :** The attacker compromises the legitimate account of an employee and then uses that account to request payments or sensitive information from various departments.

4. **Attorney Impersonation :** The attacker pretends to be lawyer, police or some other legal authority and demands bribe, pending tax payments or challans for penalties. She scares the victim of the legal action if the victim does not respond or carry out the desired action within the given time period.

5. **Data Theft :** The employees in payroll, HR or other people management departments are targeted to obtain the personal information of company employees, suppliers and contractors. This information can then further be used to victimise them impersonating as coming from the company legitimately.

3. Malicious Attachments

Emails provide an easy way to send various types of files along with the email content.

✍ ***Definition** : The files, containing malicious information or executable programs such that to harm the receiver, when sent along with the email are called malicious attachments.*

- The email might claim various things to provoke you to download and open the attachments. Following are some of the claims that the emails usually make when sending malicious attachments.
  1. Your bank account is ready. Please find the details attached.
  2. Your credit card application is approved. Please find the details attached.
  3. Your parcel is on the way. Please find the details attached.
  4. Your email was hacked. Please find the details attached.
  5. You have won a paid trip to Singapore. Please find the details attached.
  6. Your resume is shortlisted for various companies. Please find the details attached.
  7. The critical software update is required to secure your computer. Please find the details attached.
  8. The sick child needs your help immediately. Please find the details attached.
  9. Your last email was not delivered. Please find the details attached.
  10. You have won a lottery. Fill the attached form to claim the prize money.
- There could be several other email subjects and body that might get you interested. Do not open these attachments.
- The malicious attachments could be
  1. Documents containing macros that can execute the macro code when opened.
  2. Binary files or executable programs that can be run when clicked.
  3. Malware that might get installed.
  4. Simple forms that might collect your personal information.
  5. Documents containing links that take you to malicious sites.
  6. Documents spreading fake news or fear.
- Some of the attachments could be purposely sent as zipped or as archive files (so that you cannot preview the attachment content) that when unzipped could do several things as part of the unzip process. The zipped files, when unzipped, could consume a significant amount of storage space on the user's system and may cause DoS attack as well.

4. Email Bombing

✍ ***Definition** : Email bombing is a DoS attack where the attacker sends a huge volume of emails to a particular mailbox to cause overflow.*

- Email bombing could involve many large emails (say with attachments) or several small emails. The objective of email bombing is to cause DoS attack for the user such that she is not able to use the email service or access her mailbox.
- With such a huge volume of emails in the user's mailbox, she may not be able to receive new messages (each user usually has the quota of emails that she can get) or even if she could get new legitimate emails, she may not be able to find them amongst several junk emails. Such a huge number of emails could potentially crash the email server as well.

5. Unsolicited Bulk Email (UBE)

✍ ***Definition :*** *Unsolicited Bulk Email (UBE) are email messages that are randomly sent in bulk and contain unwanted, irrelevant, inappropriate or malicious content.*

- These are also called SPAM messages. UBEs are mostly sent randomly without targeting a specific group. In most cases, they are used as the first targeting and filtering technique for identifying the potential future victims who are most likely to open such randomly sent emails. These victims are then sent specific phishing emails.
- UBEs, coming from several email domains, can potentially fill your mailbox and may make it difficult for you to work with legitimate emails. However, most of the email providers have various techniques to separate out these emails in a separate folder for your future review and such emails are less likely to enter your inbox directly. However, these emails still take up your mailbox storage capacity, processing power, clean-up time and hence are distracting.
- The UBE sending techniques could be categorised as following.

1. **Email appending :** Email appending is a technique for sending marketing UBEs by matching a vendor's customer data with email addresses. A lot of companies have huge databases containing demographic details and the corresponding email addresses. The vendors can buy such databases and can send marketing emails to a wide set of users.
2. **Image-based UBEs :** Image-based UBEs embed text into a set of images and just send those images as email messages. The technique is adopted to circumvent (avoid) spam filters that mostly work on text content in the spam emails. The spam filters usually cannot read texts in images and may not classify the incoming email as spam solely based on the email content. However, the spam filters still can look at various heuristic (historical trends) data and determine if a particular email message could likely be a spam. Such heuristic parameters could be source email address, email subject, time and day of email sent, the image size, etc.
3. **Blank UBEs :** The blank UBEs do not have any email body content and are sent just with a dummy email subject such as "no subject". The objective of sending blank UBEs is to validate an email address' existence. The attacker might get a database of various email addresses. She sends a blank UBE to all of those email addresses and depends upon the email delivery bounce failure messages to separate out valid email addresses from invalid email addresses. The attacker can then send other phishing or spam emails to valid addresses only and avoid spending time and energy working with invalid email addresses.

   Some spam email messages may appear to be blank but may not actually be blank. The attacker could use transparent images or text. Opening of such emails may directly contact the attacker's server and may use the web bug technique for tracking you or downloading malicious content.
4. **Backscatter UBEs :** Backscatters are incorrectly sent bounce messages by the email servers. The email server may be misconfigured to send bounce messages when it fails processing of incoming spam emails.

6. DoS against Email Servers

Like any other server, the email server requires protection from DoS attacks. If the email server is unavailable, the users will not be able to send or receive emails. You should follow the regular DoS protection mechanisms such as capacity planning, firewalls, high availability to ensure the availability of the email servers. The attacker, instead of attacking the individual users, could potentially attack the email server to cause more severe and widespread attack on email services.

7. Unauthorised Email Senders

✍ ***Definition :*** *Unauthorised email senders appear to be legitimate senders but are using the organisation's email domain or the IP address of the email domain maliciously.*

- The unauthorised email senders may impersonate as someone legitimate from the organisation and may initiate fraudulent activities such as money transfers or soliciting sensitive information. It is difficult to block such senders or mark them as spammers.
- The unauthorised senders are possible in the following situations.

1. A malware may be present on the employee's system and may be sending emails without her knowledge.
2. A malicious employee herself might be operating an email server without organisation's knowledge or approval.
3. Various connected devices in the organisation, such as printers and IP-based CCTV cameras, might be hijacked for sending emails.

8. Unauthorised Email Receivers

✍ ***Definition*** *: Unauthorised email receivers appear to be legitimate receivers but are using the organisation's email domain or the IP address of the email domain maliciously.*

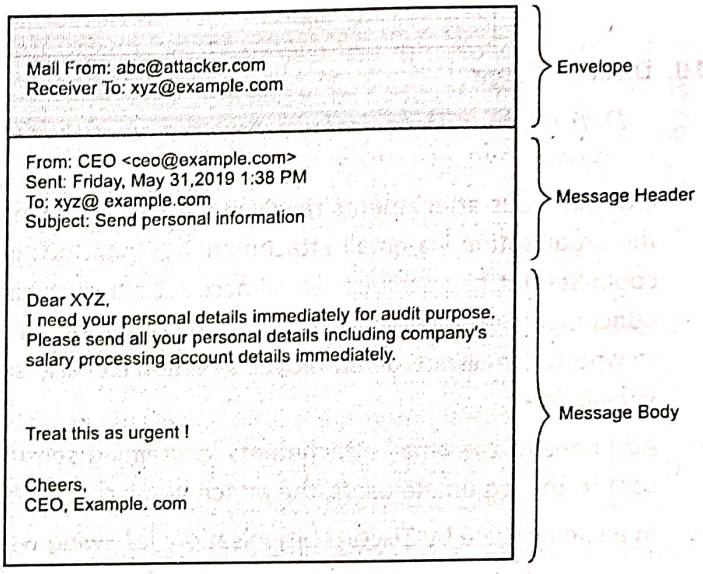
- Unauthorised email receiver could be an entry point in the organisation's network for malicious emails. These receivers may not be guarded and hence could be missed out from the regular email filters and scanning before being allowed on the organisation's network. Such emails can then install malware or spread the email to others in the organisation.
- The unauthorised receivers are possible in the following situations.

1. Obsolete, test or sample email addresses are not deleted and are still open to receiving emails.
2. An employee uses other email services, in the organisation's network that the organisation does not have control on.
3. Various connected devices in the organisation, such as printers and IP-based CCTV cameras, might be hijacked for receiving emails and malicious attachments.

9. Email Address Spoofing

✍ ***Definition*** *: Email address spoofing is a technique where the attacker can trick the email receiver to believe that the email came from a legitimate sender. It is very common.*

- It is very common. Any email primarily consists of three sections.
  1. The envelope
  2. The message header
  3. The message body
- Usually, the envelope section is hidden from the user and is filled automatically by the email server when the email is sent or received. The user sees the header section and the attacker can potentially change the information contained in the header section.



- The email protocol (SMTP) allows you to put any name in the "FROM" field in the header section. The attacker manipulates the email header information to make it look like coming from a legitimate sender. Apart from the "FROM" field, the attacker can also manipulate REPLY-TO and RETURN-PATH addresses. The SMTP protocol does not provide an address authentication mechanism by default. So, when the user receives the email, she checks the header section and trusts it and carries out the malicious actions intended by the attacker.
- Email address spoofing is a common technique used for Phishing or SPAM emails. The goal is to make the receiver believe that the email can be trusted to come from a valid source and the desired action mentioned in the email can be taken.

- Here is a sample email with spoofed "FROM" field.

amazon

From: Amazon < amazonpay@amz.com>
Sent: Friday, May 31, 2019 1.38 PM
To: Rishi Kumar
Subject: Amazon account password reset

The receiver does not carefully check the email address and trusts the user name shown in the FROM Field.

Dear Rishi,
At Amazon, we take customer safety very seriously. Consequently, we have improved our password policies to require strong passwords.
We have suspended your account and you require to set a new password to continue to use it.
Your new password must include

- At least 12 characters
- One digit
- One special character
- One uppercase alphabet

Please click on the below link and reset your account password now.
Reset your amazon password now.

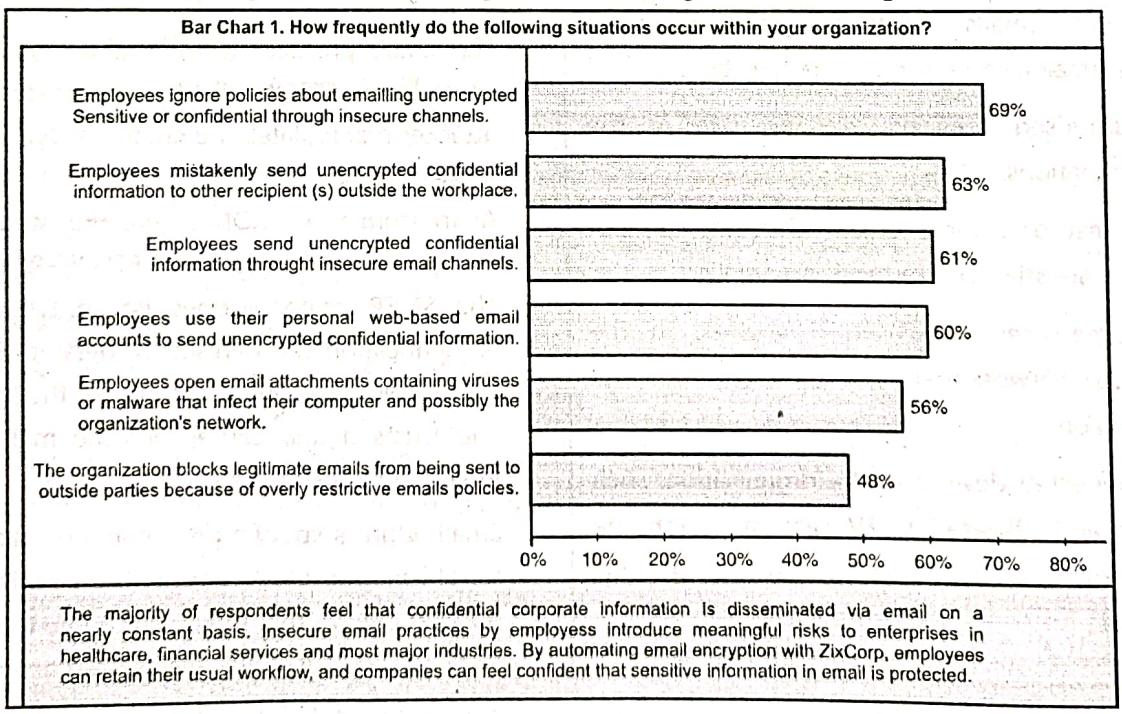
You are reminded that never share your amazon account details, bank details or OTP with anyone.

Yours sincerely,
Amazon
https://amazon.com

10. Data Leakage

✍ ***Definition** : Data leakage involves unauthorised forwarding or transmission of data, from within the organisation, externally.*

- Like malicious attachments received via emails within the organisation, the sensitive information might go outside the organisation via email attachments. A malicious insider can forward the information directly to the attacker or could send it her personal email account citing excuses such as working from home or taking backup. The email attachments originating from within the organisation are seldom checked whether they are allowed to be sent and to whom. A malware on employee's system can also send emails containing sensitive information without the user's knowledge.
- Additionally, the email attachments, containing sensitive information, might not be encrypted. So, even if they are sent to the legitimate users, the attachments can still be extracted.
- In a survey done by Zixcorp organisation, following were the findings for data leakage over email.



11. Email Content Leakage

✍ ***Definition :*** *Email content leakage is similar to data leakage but more specifically referring to the information contained in the email body or email account rather than sensitive attachments.*

Email content can be leaked in the following scenarios.

1. The attacker may be able to collect email message packets over the network via eavesdropping attack.
2. A malware may make copies of emails sent or received on the user's device and forward them to the attacker.
3. The attacker can carry out phishing, social engineering or other types of email based attacks to get the sensitive information from the users directly.
4. The attacker may also do traffic analysis and infer the sensitive information. For example, frequent emails between the HR department and senior management might mean employee layoffs.
5. The attacker can compromise the email account and get access to all the emails. People usually save important emails in the email account. A compromise of the email account would mean disclosure of such sensitive emails.
6. Your email service provider might suffer a data beach. Such data breach could expose several email account details that the attacker can use to get access to the sensitive emails. One of the biggest data breaches was for Yahoo where approximately 3 billion user accounts were exposed!

12. Tampering with Email Content

✍ ***Definition :*** *Tampering with email content involves altering the information contained in the email body.*

Like any other message travelling over the internet, email messages could be tampered as well to alter the information contained therein. The email, when received, might look like being sent from the original sender with the content that you read but the attacker might have altered its content.

Email content can be tampered in below scenarios.

1. The email message packets are replayed after modifying the information.
2. The email message received by the attacker can be text-edited and re-transmitted to other users using email address spoofing technique.
3. The attacker could compromise the email account and alter the draft email messages that are yet to be sent out or can compose new emails and send.
4. The email service provider could add additional promotional content as part of each email message sent o received using its service.

13. Email Server Security, Software and Protocols

- Email service is dependent on email server, email client, other software such as spam filters and various protocols. It is a complex system and involves several components. It is difficult to guarantee end to end email security without adequately protecting and configuring all the interdependent components.
- There could be various vulnerabilities discovered overtime which must be fixed to keep the email services secure and operational. Emails also involve users which are always the weakest link in security and might require extensive training to understand complexities of email security. A variety of email attacks can be formulated exploiting the vulnerabilities in email software, protocols and people.

Countermeasures against Email Attacks

- You should refer to the section "Countermeasures against Social Engineering (and Phishing)" for understanding the general recommendations around email security.
- However, protection against email attacks requires a holistic approach instead of isolated quick fixes. NIST Special Publication 800-45 has listed several guidelines on email security. It is beyond the scope of this book to discuss each of these in detail, but they are listed here for your reference. I would suggest

downloading a copy of this publication and go through it to understand several recommendations around email security. You could also refer to NIST Special Publication 800-177 that detail about setting up Trustworthy Email service.

– Here are the security recommendations from NIST SP 800-45.

1. Planning and Managing Mail Servers

– Plan the installation and deployment of mail server

i. Identify functions of the mail server

ii. Identify categories of information that will be stored on, processed on, and transmitted through the mail server

iii. Identify security requirements of information

iv. Identify requirements for continuity of mail services

v. Identify a dedicated host to run the mail server

vi. Identify network services that will be provided or supported by the mail server

vii. Identify users and categories of users of the mail server and determine privilege for each category of user

viii. Determine how the mail server will be managed (e.g., locally, remotely)

ix. Identify user authentication methods for the mail server

x. Identify security or privacy requirements for email address-related information

– Choose appropriate operating system for mail server

i. Minimal exposure to vulnerabilities

ii. Ability to restrict administrative or root level activities to authorised users only

iii. Ability to deny access to information on the server other than that intended to be available

iv. Ability to disable unnecessary network services that may be built into the operating system or server software

v. Ability to log appropriate server activities to detect intrusions and attempted intrusions

vi. Availability of trained, experienced staff to administer the server and server products

– Plan the location of the mail server

i. Appropriate physical security protection mechanisms

ii. Appropriate environmental controls to maintain the necessary temperature and humidity

iii. Backup power source

iv. Preparation for known natural disasters

– Patch and upgrade operating system

i. Create and implement a patching process

ii. Identify, test, and install all necessary patches and upgrades to the operating system

– Remove or disable unnecessary services and applications

i. Use separate hosts for Web servers, directory servers, and other services

– Configure operating system user authentication

i. Remove or disable unneeded default accounts and groups

ii. Disable non-interactive accounts

iii. Create the user groups for the particular computer

iv. Create the user accounts for the particular computer

v. Check the organisation's password policy, and set account passwords appropriately (e.g., length, complexity)

vi. Configure computers to prevent password guessing

vii. Install and configure other security mechanisms to strengthen authentication

– Configure resource controls appropriately

i. Set access controls for files, directories, devices, and other resources

ii. Limit privileges for most system-related tools to authorised system administrators

– Select, install, and configure additional software to provide needed controls not included in the operating system

- Test the security of the operating system

i. Test operating system after initial install to determine vulnerabilities

ii. Test operating system periodically to determine new vulnerabilities

2. Securing Mail Servers and Content

- Harden the mail server application

i. Install the mail server software on a dedicated host (if Web-based mail access is desired, install the mail server software on a different host from the Web server)

ii. Apply any patches or upgrades to correct for known vulnerabilities

iii. Create a dedicated physical disk or logical partition (separate from operating system and mail server application) for mailboxes, or host the mailboxes on a separate server

iv. Remove or disable all services installed by the mail server application but not required (e.g., Web-based mail, FTP, remote administration)

v. Remove or disable all unneeded default login accounts created by the mail server installation

vi. Remove all manufacturer documentation from server

vii. Remove any example or test files from server

viii. Apply appropriate security template or hardening script to the server

ix. Reconfigure SMTP, POP and IMAP service banners (and others as required) NOT to report mail server and operating system type and version

x. Disable dangerous or unnecessary mail commands (e.g., VRFY and EXPN)

- Configure operating system and mail server access controls

i. resources

ii. Limit the access of users through additional access controls enforced by the mail server, where more detailed levels of access control are required

iii. Configure the mail server application to execute only under a unique individual user and group identity with restrictive access controls

iv. Ensure the mail server is not running with root or system/administrator privileges

v. Configure the host operating system so that the mail server can write log files but not read them

vi. Configure the host operating system so that temporary files created by the mail server application are restricted to a specified and appropriately protected subdirectory

vii. Configure the host operating system so that access to any temporary files created by the mail server application is limited to the mail server processes that created these files

viii. Ensure that the mail server cannot save files outside of the specified files structure dedicated to the mail server

ix. Configure the mail server to run in a chroot jail on Linux and Unix hosts

x. Install users' mailboxes on a different server (preferred), hard drive, or logical partition than the operating system and mail server application

xi. Configure the mail server application so it cannot consume all available space on its hard drives or partitions

xii. Limit the size of attachments that are allowed

xiii. Ensure log files are stored in a location that is sized appropriately

- Protect email from malware

i. Determine which types of attachments to allow

ii. Consider restricting the maximum acceptable size for attachments

iii. Determine if having access to personal email accounts from organisational computers is appropriate

iv. Determine which types of active content should be permitted within email messages

v. Implement centralized malware scanning (on the firewall, mail relay, mail gateway, and/or mail server)

vi. Install malware scanners on all client hosts

vii. Implement centralized content filtering

viii. Configure content filtering to block or tag suspicious messages (e.g., phishing, spam)

ix. Configure content filtering to strip suspicious active content from messages

x. Configure lexical analysis if required

xi. Take steps to prevent address spoofing, such as blocking emails from external locations using internal "From" addresses

xii. Create a security policy that addresses content filtering

xiii. Have the security policy reviewed by appropriate legal, privacy, and human resources authorities

xiv. Add a legal disclaimer to emails, if required

xv. Educate users on the dangers of malware and how to minimize those dangers

xvi. Notify users when an outbreak occurs

– Block spam-sending servers

i. Configure mail gateways or firewalls to use LDAP lookup to confirm the existence of email recipients

ii. Configure mail server to block email from open relay blacklists or DNS blacklists, if required

iii. Configure mail server to block email from specific domains, if required

– Configure authenticated mail relay on the server

– Configure mail server to use encrypted authentication

– Configure mail server to support Web access only via SSL/TLS and only if such access is deemed necessary

3. Implementing a Secure Network Infrastructure

– Mail server is located on the internal network and protected by a mail gateway and/or firewall, or Mail server is located in a DMZ

– Firewall configuration

i. Mail server is protected by a firewall

ii. Mail server, if it faces a higher threat or if it is more vulnerable, is protected by an application layer firewall

iii. Firewall controls all traffic between the Internet and the mail server

iv. Firewall blocks all inbound traffic to the mail server except the necessary ports, such as TCP ports 25 (SMTP), 110 (POP3), 143 (IMAP), 398 (LDAP), 636 (secure LDAP), 993 (secure IMAP), and 995 (secure POP)

v. Firewall blocks (in conjunction with the intrusion detection or prevention system) IP addresses or subnets that the IDS or IPS reports are attacking the organisational network

vi. Firewall blocks known "blacklisted" networks or subnets, as identified by a trusted external security response center

vii. Firewall notifies the network administrator or mail server administrator of suspicious activity through an appropriate means

viii. Firewall provides content filtering and malware scanning

ix. Firewall is configured to protect against DoS attacks

x. Firewall logs critical events

xi. Firewall and firewall operating system patched to latest or most secure level

– Intrusion detection and prevention systems

i. IDPS configured to monitor traffic network traffic to and from the mail server

ii. IDPS configured to monitor changes to critical files on mail server (host-based IDPS or file integrity checker)

iii. IDPS configured to monitor the system resources available on the mail server host (host-based IDPS)

iv. IDPS blocks (in conjunction with the firewall) IP addresses or subnets that are attacking the organisational network

v. IDPS notifies the necessary parties of suspected attacks through appropriate means according to the

organisation's incident response policy and procedures

vi. IDPS configured to maximize detection with an acceptable level of false positives

vii. IDPS configured to log events and to capture packet header information for network events

viii. IDPS updated with new attack signatures frequently (e.g., on a daily to weekly basis, typically after testing the updates)

– Network switches

i. Network switches are used to protect against network eavesdropping

ii. Network switches are configured in high-security mode to defeat ARP spoofing and ARP poisoning attacks

iii. Network switches are configured to send all traffic on network segment to network-based IDPS

4. Securing Mail Clients

– Patch and update mail clients

i. Update mail client to newest or most secure version

ii. Apply any necessary patches to mail client (in conformance with organisational polices and configuration management)

iii. Apply any necessary patches to Web browser (for mail clients that are integrated with browser)

– Configure mail client security features

i. Disable automatic message preview

ii. Disable automatic opening of messages

iii. Disable automatic loading of pictures in messages

iv. Disable downloading and processing of active content (if appropriate)

v. Enable anti-spam and anti-phishing features

vi. Reconfigure portable mail clients, such as those on cell phones and PDAs, to improve their security

vii. Ensure that security policy supports the protection of portable mail clients, such as requiring anti-virus software to be installed and enabled

viii. Limit access to VPN clients and other remote access applications on mobile devices, or remove the clients/applications if they are not needed

– Configure authentication and access

i. Enable secure authentication and access

ii. Disable ability of mail client to store username and passwords

iii. Configure client to use encryption (TLS) for SMTP, POP, and IMAP communications

iv. Set restrictions on the selection of email addresses, such as ensuring they are unrelated to user account names

– Secure the mail client host operating system

i. Keep the OS updated to the most secure patch level

ii. Configure the OS to allow only the appropriate user(s) to access locally stored messages and mail client configuration files

iii. Secure or remove Windows Script Host (Windows hosts only)

iv. Change the default action on files associated with the Windows Script Host from execute to edit (Windows hosts only)

v. Ensure that the OS is configured to show full file extensions (Windows hosts only)

vi. Install an anti-virus application and configure it to scan incoming messages and attachments; also install an anti-spyware application if the anti-virus software does not offer robust anti-spyware capabilities

vii. Install a personal firewall if needed to protect the computer from unauthorised communications

viii. Ensure the OS enforces the concept of least privilege, because malicious code runs in the security context on which it was launched (i.e., the user's access level)

ix. Ensure that critical components of the operating system are protected from malicious code

x. Use a file encrypting application to protect the email stored locally on the user's hard drive (especially important for mobile devices)

xi. Configure the OS to automatically lock the current session after a fixed period of inactivity

– Provide security for email message content (e.g., S/MIME, OpenPGP)

– Enable and install only absolutely necessary plug-ins from trusted sources

– Access to Web-based mail systems

i. Configure Web-based mail access to only use 128-bit SSL/TLS connections

ii. Make users aware of what they should do before granting them access to Web-based mail

5. Administering the Mail Server

– Logging

i. Log IP stack setup errors

ii. Log resolver configuration problems (e.g., DNS, NIS)

iii. Log mail server configuration errors (e.g., mismatch with DNS, local configuration error, out-of-date alias database)

iv. Log lack of system resources (e.g., disk space, memory, CPU)

v. Log alias database rebuilds

vi. Log logins (failed, and also successful if adequate space is available)

vii. Log security problems (e.g., spamming)

viii. Log lost communications (network problems)

ix. Log protocol failures

x. Log connection timeouts

xi. Log connection rejections

xii. Log use of VRFY and EXPN commands

xiii. Log send on behalf of

xiv. Log send as

xv. Log malformed addresses

xvi. Log message collection statistics

xvii. Log creation of error messages

xviii. Log delivery failures (permanent errors)

xix. Log messages being deferred (transient errors)

xx. Store logs on a separate logging server

xxi. Backup and archive logs according to organisational requirements

xxii. Review logs daily

xxiii. Review logs weekly (for more long-term trends)

xxiv. Use automated log file analysis tool(s)

– Mail server backups

i. Create a mail server backup policy

ii. Back up mail server differentially or incrementally on a daily to weekly basis

iii. Back up mail server fully on a weekly to monthly basis

iv. Periodically archive backups

– Recovering from a compromise

i. Report incident to organisation's computer incident response capability

ii. Isolate compromised system(s) or take other steps to contain attack so additional evidence can be collected

iii. Consult, as appropriate, with management, legal counsel, and law enforcement expeditiously

iv. Investigate similar hosts to determine if the attacker has also compromised other systems

v. Analyse the intrusion

vi. Restore the system

vii. Test system to ensure security

viii. Reconnect system to network

ix. Monitor system and network for signs that the attacker is attempting to access the system or network again

x. Document lessons learned

– Security testing

i. Periodically conduct vulnerability scans on mail server and supporting network

ii. Update vulnerability scanner before testing

iii. Correct any deficiencies identified by the vulnerability scanner

iv. Conduct penetration testing on the mail server and the supporting network infrastructure

v. Correct deficiencies identified by penetration testing

– Remote administration

i. Use a strong authentication mechanism (e.g., public/private key pair, two factor authentication)

ii. Restrict which hosts can be used to remotely administer the mail server by IP address or by authorised users

iii. Use secure protocols (e.g., SSH, HTTPS) that can provide encryption for both passwords and data

iv. Enforce the concept of least privilege on remote administration (e.g., attempt to minimize the access rights for the remote administration accounts)

v. Change any default accounts or passwords for the remote administration utility or application

vi. Do not allow remote administration from the Internet unless mechanisms such as VPN are used

vii. Do not mount any file shares on the internal network from the mail server and vice versa

6.8 Firewalls

– Your computer is connected to the internet. How do you protect it from someone trying to access it over the internet? How do you prevent some rogue programs on your computer to send information to the attacker? Firewalls could be a mechanism.

✍ ***Definition :*** *Firewalls are network security systems that protect the computing resources on a trusted network from unauthorised access.*

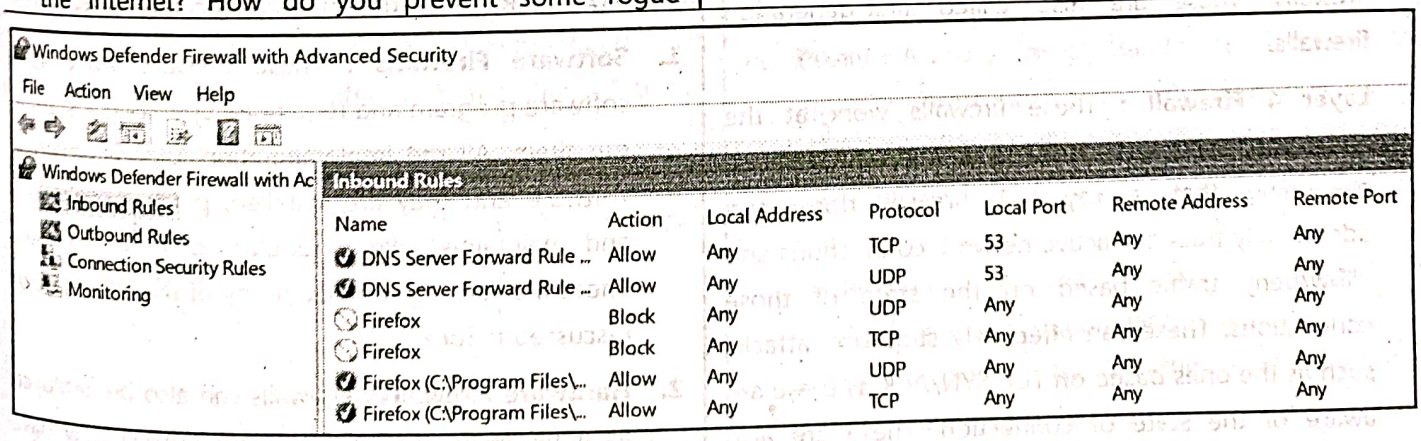
– For example, you can access google.com website but not it's webserver's operating system. If you try connecting to the webserver except over the HTTP or HTTPS, the connection would be denied. That's what a firewall does at a high level.

– You need to define various rules, as per your security requirements, in the firewall and the firewall evaluates those rules before granting or denying access to the requested resource.

6.8.1 Components of a Firewall Rule

Typically, a firewall rule consists of the following parameters –

1. Source IP address or hostname
2. Destination IP address or hostname
3. Source Port number
4. Destination Port number
5. Direction of communication [inbound or outbound]
6. Protocol name [TCP, UDP, ICMP or various others]
7. Action [allow, deny, log, etc.]
8. Various optional parameters such as Rule Name, Evaluation Order, etc.



This is a snapshot of Microsoft® Windows® Firewall.

6.8.2 Classification of Firewalls

Firewalls can be classified based on various attributes. Let's learn about their types.

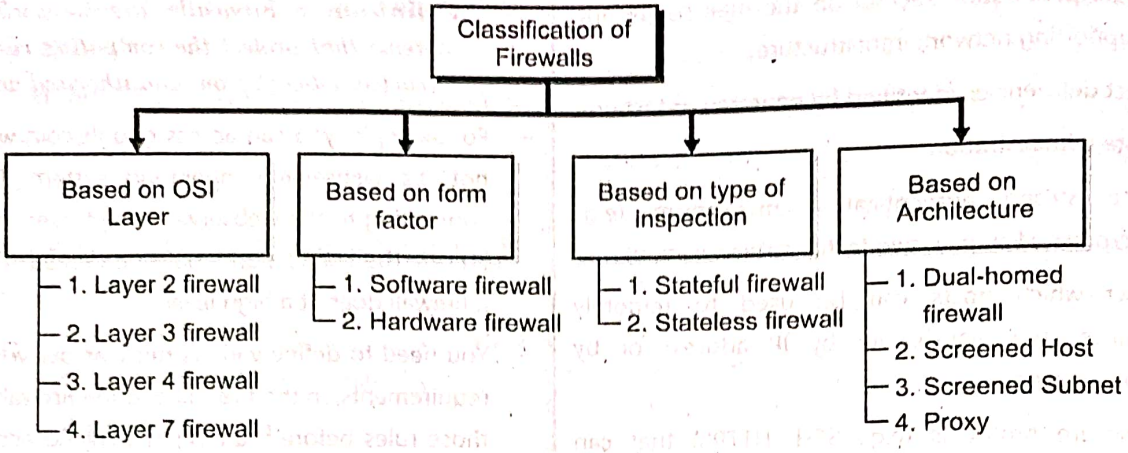


Fig. 6.8.1

A. Based on the OSI Layer

As you understand, OSI is a conceptual networking model. Based on the various layers, firewalls can be classified as following:

1. **Layer 2 Firewall :** These firewalls work at the "Data Link" layer of the OSI model. These firewalls require MAC, VLAN or device hardware level information to operate. One of the greatest advantage of these types of firewalls is that they are not IP dependent.

2 **Layer 3 Firewall :** These firewalls work at the "Network" layer of the OSI model. These filter traffic based on source/destination IP, port, and protocol. These are one of the most prevalent types of firewalls in use today. These are also called as Stateless firewalls. These are also called first-generation firewalls.

3. **Layer 4 Firewall :** These firewalls work at the "Transport" layer of the OSI model. These firewalls do everything that a Layer 3 firewall does and additionally track the active network connections and allow/deny traffic based on the state of those connections. These can effectively stop DoS attacks such as the ones based on TCP SYN/ACK as these are aware of the state of connection. These are also called as Stateful firewalls. These are also called second-generation firewalls.

4. **Layer 7 Firewall :** These firewalls are called Layer 7 but can work at three layers – Session, Presentation and Application. For simplicity, these are just called Layer 7 firewalls. Layer 7 firewalls do everything that a Layer 4 firewall does and additionally include the ability to intelligently inspect the contents of the network packets passing through them. For example, a Layer 7 firewall could deny all the HTTP requests from Korean IP addresses. They have the actual packet content level visibility and are the most advanced types of firewall in use today. These are also called third-generation firewalls.

B. Based on the Form Factor

Form factor or the footprint is the way the firewall is actually packaged and deployed. They can be classified as

1. **Software Firewalls :** These firewalls work as a software program and require an operating system to run them. All the implementation logic is coded in software and they are installed, patched, upgraded and maintained like a regular computer software. These firewalls could work at any of the OSI layers as discussed before.

2. **Hardware Firewalls :** Firewalls can also be deployed as a hardware device. Hardware firewall may have better performance and they come packaged in a ready to use hardware device. Like any other firewall,

you need to configure it as per your security requirements.

C. Based on the Type of Inspection

Firewalls can keep track of connections or just work based on the configured rules. Based on their inspection types, these can be classified as

1. **Stateful Firewalls :** These firewalls keep track of the state of connections apart from the defined firewall rules. These precisely understand various handshake protocols and can effectively stop attacks that try to manipulate connection establishment or maintenance process.
2. **Stateless Firewalls :** Stateless firewalls typically work at the Layer 3 and take decisions based on the defined rule parameters such as IP, Port and Protocol. These do no track connection states and cannot effectively protect against attacks that manipulate connection processes.

D. Based on Architecture

Firewalls can be deployed in many ways. The have special properties that make them suitable for one deployment type over the another. Based on the deployment possibilities, firewalls can be classified as

1. **Dual-homed Firewalls :** A Dual-Homed Firewall has two interfaces - one facing the external network and the other facing the internal network. It receives the external packets on one of its interfaces, evaluates firewall rules, and passes on the traffic to the designated internal resources via the second interface. The two interfaces are kept separate to isolate the external traffic with the internal traffic physically.

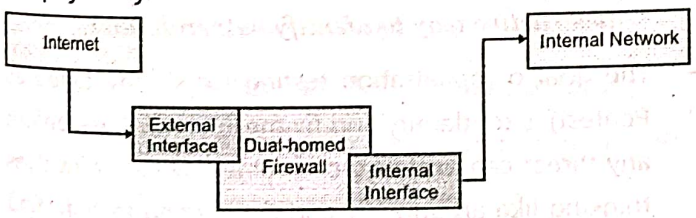


Fig. 6.8.2

2. **Screened Host :** In a screened host firewall, all internet (and other regulated) traffic goes through the firewall, no matter what. The internet router device first screens (filters) all the packets that are relevant to the network and then passes it to the Screened Host firewall for further inspection and applying rules.

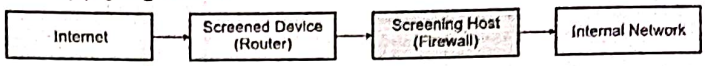


Fig. 6.8.3

3. **Screened Subnet :** In screened subnet architecture, two firewalls are used. One just after the external network and the one just before the internal network. Any network that lies between the two firewalls is called a Demilitarized Zone (DMZ). You place your public facing servers such as webservers, email servers etc. in DMZ. An attacker would have to bypass both the firewalls before she can hit the internal network. This kind of architecture is commonly used in the industry today.

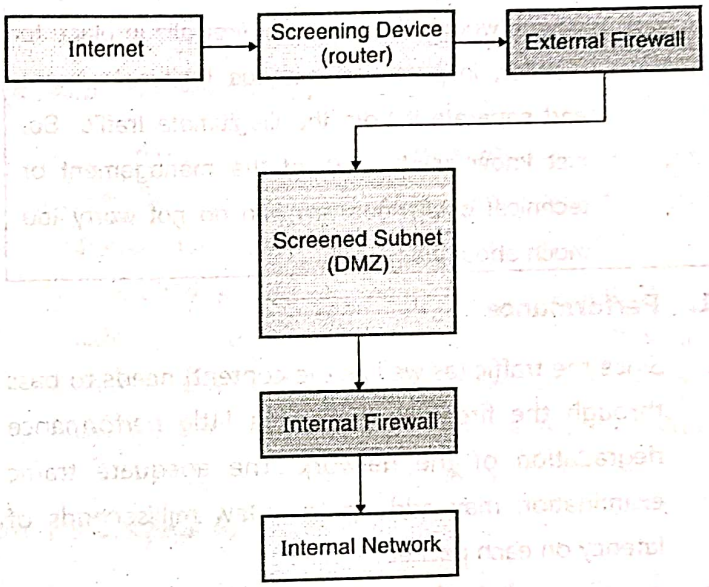


Fig. 6.8.4

4. **Proxy :** A proxy firewall stands between the trusted and the untrusted network and takes allow or deny decisions after careful inspection of what is being passed along. Like a regular proxy, the proxy firewall breaks the connection between the source and the destination. After examining the traffic, it self-establishes a connection with the destination and passes the intended traffic to the destination as if the packets were originating from it.

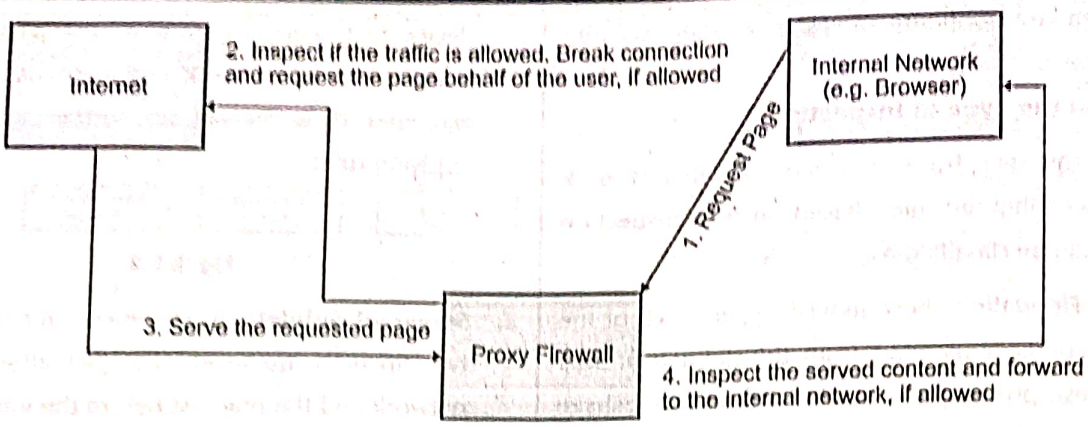


Fig. 6.8.5

6.8.3 Challenges in Managing and Deploying Firewalls

Note : Irrespective of what challenges or limitations firewalls may have, they are heavily used throughout the industry. You cannot just imagine any network without several firewalls in place to monitor, inspect and manage legitimate traffic and separate it from the illegitimate traffic. So, just know what some of the management or technical challenges are and do not worry too much about them.

1. **Performance**

   Since the traffic (as well as the content) needs to pass through the firewalls, there is a little performance degradation of the network. The adequate traffic examination may add up to a few milliseconds of latency on each packet.

2. **Business agility**

   Firewall rules are usually manually added, edited or deleted. The pace of business might be too high to require several changes to the firewall rules frequently. Keeping up with these changes without making errors is difficult.

3. **Costs**

   Modern (or advanced) firewalls that provide content and protocol level inspection may be cost-prohibitive for small or medium sized organisations.

4. **Insider attacks**

   Firewalls are usually designed and deployed to protect a trusted network from an untrusted network. But, if there were other vulnerabilities (such as a missing OS security patch) that were exploited such that an attacker is already on the trusted network, firewalls might not be able to protect or limit damages to the other resources on the trusted network.

5. **Managing firewalls themselves**

   Like your OS, printers or other software or hardware devices, firewalls need to be installed, patched, updated, etc. to remain operational. This adds a management overhead. Additionally, firewalls could have known vulnerabilities that need to be patched else a firewall that itself is lacking protection may not be very useful in providing you the required level of protection.

6.9 Penetration Testing

✍ ***Definition** : Penetration testing involves carrying out attacks on systems in a non-destructive way to identify vulnerabilities.*

– The goal of penetration testing (or shortly called as Pentest) is to identify and fix the vulnerabilities before any threat can exploit those vulnerabilities. It involves thinking like an attacker and conducting tests to find out vulnerabilities in the system. These tests are carried out by system experts who are deemed to be as skilled as the attacker. They are part of a team which is usually called as **Red Team.**

***Definition** : In the industry, the term "Red Team" is generally used for any team that focuses on identifying improvement opportunities in the organisation.*

Conducting penetration testing involves the following high-level steps as shown in Fig. 6.9.1.

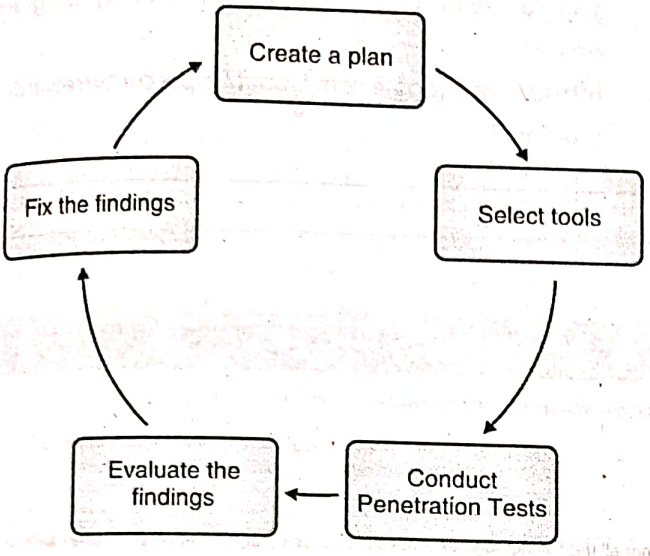


Fig. 6.9.1

(i) Create a plan

- In this step, you first seek management approval. Getting approval is crucial to ensure that your tests are not considered unwanted and unexpected and you are not charged with any liabilities for damages, if and when occurred. Once the approval is in place, you define a scope for the test.
- The scope should include
  (a) Systems to be tested
  (b) Specific dates and time
  (c) Total time duration for which tests would be performed
  (d) Any business risks involved
  (e) Deliverables to be produced as part of testing
  (f) Should you test with security controls turned off or on.

(ii) Select tools

- Once the scope is identified, you need to select tools specific to the type of systems you are going to test and the nature of the tests that you intend to carry out on those systems. The penetration testing tools have various capabilities. It is important to thus choose the right tool based on the system and the nature of test desired.
- For example, password cracking tools are different from network packet capturing tools. If your penetration test requires a network packet capture tool, you would include that tool in your list of tools required for penetration testing exercise at hand. Some of the common tools are Cain and Abel, Metasploit, Nexpose, Wireshark and Nmap.

(iii) Conduct penetration tests

At this step, you conduct several tests as per your plan and scope. The tests could take several weeks. The goal is to discover as many vulnerabilities as you can within the given duration of test and report them with their potential impact on the systems under test.

(iv) Evaluate the findings

Once the vulnerabilities have been identified, you need to carefully evaluate the impact those vulnerabilities can have on the systems. Each vulnerability should be prioritized according to its impact. A penetration testing report should then be generated describing the systems, tests conducted on those systems and the test outcomes (vulnerabilities and any other security related observations).

(v) Fix the findings

- The findings from penetration testing are then fixed in order of their impact. Fixing the findings is a crucial step to ensure that the attacker cannot exploit them.
- Regular penetration testing may reveal several vulnerabilities beforehand and can help mitigate the risk of cyberattacks. Penetration testing is highly recommended for all internet facing servers, websites, portal or applications. These days quite a few companies offer a bug bounty program to identify vulnerabilities before they are widely abused or exploited.

✍ ***Definition :*** *A bug bounty program is an open offer from a company for anyone in the world from anywhere to conduct non-destructive penetration testing on its systems (websites, portals, applications, etc.) and report the findings securely to it for appropriate fixing and handling of those findings. The reporter is awarded and paid in proportion of the impact that the reported finding could have on the system and the company's business.*

- The bug bounty program leverages (gets access to) worldwide talent pool of experts who can help the company to prevent cyberattacks. In return the reporters get recognition and compensation for their work from the company.

  You can read about Google's bug bounty program on https://www.google.com/about/appsecurity/reward-program/.

Google Application Security

Home Learning Reward Programs Hall of Fame Research

Google VRP Patch Rewards AutoFuzz Patch Rewards Research Grants Chrome Rewards Android Rewards Google Play Rewards

Google Vulnerability Reward Program (VRP) Rules

We have long enjoyed a close relationship with the security research community. To honor all the cutting-edge external contributions that help us keep our users safe, we maintain a Vulnerability Reward Program for Google-owned web properties, running continuously since November 2010.

Services in scope

In principle, any Google-owned web service that handles reasonably sensitive user data is intended to be in scope. This includes virtually all the content in the following domains:

- *.google.com
- *.youtube.com
- *.blogger.com

Bugs in Google Cloud Platform, Google-developed apps and extensions (published in Google Play, in iTunes, or in the Chrome Web Store), as well as some of our hardware devices (Home, OnHub and Nest) will also qualify. See our Android Rewards and Chrome Rewards for other services and devices that are also in scope.

On the flip side, the program has two important exclusions to keep in mind:

- **Third-party websites.** Some Google-branded services hosted in less common domains may be operated by our vendors or partners (this notably includes *zagat.com*). We can't authorize you to test these systems on behalf of their owners and will not reward such reports. Please read the fine print on the page and examine domain and IP WHOIS records to confirm. If in doubt, talk to us first!
- **Recent acquisitions.** To allow time for internal review and remediation, newly acquired companies are subject to a six-month blackout period. Bugs reported sooner than that will typically not qualify for a reward.

Qualifying vulnerabilities

Any design or implementation issue that substantially affects the confidentiality or integrity of user data is likely to be in scope for the program. Common examples include:

- Cross-site scripting,
- Cross-site request forgery,
- Mixed-content scripts,
- Authentication or authorization flaws,
- Server-side code execution bugs.

New! In addition, significant abuse-related methodologies are also in scope for this program, if the reported attack scenario displays a design or implementation issue in a Google product that could lead to significant harm.

Following is a snapshot of awards it offers for the reported security bugs (vulnerabilities).

Reward amounts for security vulnerabilities

New! To read more about our approach to vulnerability rewards you can read our Bug Hunter University article here.

Rewards for qualifying bugs range from $100 to $31,337. The following table outlines the usual rewards chosen for the most common classes of bugs:

Category	Examples	Applications that permit taking over a Google account [1]	Other highly sensitive applications [2]	Normal Google applications	Non-integrated acquisitions and other sandboxed or lower priority applications [3]
		Vulnerabilities giving direct access to Google servers			
Remote code execution	*Command injection, deserialization bugs, sandbox escapes*	$31,337	$31,337	$31,337	$1,337 - $5,000
Unrestricted file system or database access	*Unsandboxed XXE, SQL injection*	$13,337	$13,337	$13,337	$1,337 - $5,000
Logic flaw bugs leaking or bypassing significant security controls	*Direct object reference, remote user impersonation*	$13,337	$7,500	$5,000	$500
		Vulnerabilities giving access to client or authenticated session of the logged-in victim			
Execute code on the client	Web: *Cross-site scripting* Mobile / Hardware: *Code execution*	$7,500	$5,000	$3,133.7	$100
Other valid security vulnerabilities	Web: *CSRF, Clickjacking* Mobile / Hardware: *Information leak, privilege escalation*	$500 - $7,500	$500 - $5,000	$500 - $3,133.7	$100

Review Questions

Here are a few review questions to help you gauge your understanding of this chapter. Try to attempt these questions and ensure that you can recall the points mentioned in the chapter.

Web Security Considerations

Q. 1 What are the goals of Secure Socket Layer (SSL)? **(4 Marks)**

Q. 2 Give a brief overview of SSL protocol. **(6 Marks)**

Q. 3 Describe SSL Record Layer Protocol. **(8 Marks)**

Q. 4 Write a short note on SSL Alert Protocol. **(4 Marks)**

Q. 5 With a neat diagram, explain SSL Handshake process. **(8 Marks)**

Q. 6 Joe is browsing various websites and finds that some use HTTP where some use HTTPS. He is confused. What would you explain him if he approaches you? **(6 Marks)**

Q. 7 Compare HTTP and HTTPS. **(6 Marks)**

Q. 8 It is recommended to use HTTPS. Comment. **(6 Marks)**

Q. 9 What are some of the ways you can visually examine security of a website that uses HTTPS? **(4 Marks)**

Q. 10 List the various usage of Secure Shell (SSH) protocol. **(4 Marks)**

Q. 11 Write a short note on SSH Authentication Protocol. **(5 Marks)**

Q. 12 What is the function of SSH Connection Protocol? Explain. **(8 Marks)**

Q. 13 Describe the services provided by SSH Transport Layer Protocol. **(6 Marks)**

Q. 14 List a few algorithms supported by SSH Transport Layer Protocol. **(4 Marks)**

Q. 15 Draw a block diagram and list steps involved in establishing a SSH connection. **(5 Marks)**

Q. 16 With a simple block diagram, explain the steps involved in a SET-based transaction. **(6 Marks)**

User Authentication and Session Management

Q. 17 Take a real-life example and explain the problem that sessions solve from user as well as web server's perspective. **(6 Marks)**

Q. 18 What is used to create a session binding? List its characteristics. **(6 Marks)**

Q. 19 Explain in brief session lifecycle. **(4 Marks)**

Q. 20 What are the various session management mechanisms? Compare them. **(6 Marks)**

Q. 21 What is a cookie? Explain its structure and content in detail. **(8 Marks)**

Q. 22 What is Session Hijacking? Explain various techniques that could be used for hijacking sessions. **(8 Marks)**

Q. 23 Write a short note on Session Fixation. **(5 Marks)**

Q. 24 Draw a Comparison table between Session Hijacking and Session Fixation. **(6 Marks)**

Web Browser Attacks

Q. 25 Write a short note on Account Harvesting. **(4 Marks)**

Q. 26 Explain Account Harvesting Lifecyle. **(4 Marks)**

Q. 27 Explain Account Harvesting in detail and list its countermeasures. **(8 Marks)**

Q. 28 What are web bugs? Explain. **(5 Marks)**

Q. 29 With a neat block diagram, explain how web bugs work on websites? **(8 Marks)**

Q. 30 With a neat block diagram, explain how web bugs work on emails? **(8 Marks)**

Q. 31 What are web bugs? List their countermeasures. **(8 Marks)**

Q. 32 James is super happy that he has won a new iPhone. He clicked on a link and is expecting an iPhone soon. He comes to you to share this good news. You get worried hearing this. James asks you, what happened? What are you going to tell him about the reason he might have won the iPhone? **(6 Marks)**

Q. 33 With a neat diagram, explain clickjacking. **(8 Marks)**

Q. 34 What is the use of Content Security Policy (CSP) Frame-Ancestors Directive? Explain. **(8 Marks)**

Q. 35 Write a short note on X-Frame-Options Response Headers. **(4 Marks)**

Q. 36 With a neat diagram, explain how Cross-Site Request Forgery (CSRF) works. **(8 Marks)**

Q. 37 How may you use Security Tokens to provide protection against Cross-Site Request Forgery (CSRF)? **(5 Marks)**

Q. 38 How may you use HTTP Headers to provide protection against Cross-Site Request Forgery (CSRF)? **(5 Marks)**

Q. 39 Write a short note on Double Submit Cookie. **(4 Marks)**

Q. 40 What protecting is SameSite Cookie Attribute most likely to provide? **(4 Marks)**

Q. 41 Re-validation could be a simple technique to protect against Cross-Site Request Forgery (CSRF). Explain with a neat block diagram. **(6 Marks)**

Q. 42 Explain Phishing and its variants. **(8 Marks)**

Q. 43 Explain social engineering attack cycle. **(8 Marks)**

Q. 44 Explain various countermeasures against social engineering from user and organisation perspective. **(8 Marks)**

Q. 45 What is Pharming? **(6 Marks)**

Q. 46 Explain Pharming with a neat block diagram and list its countermeasures. **(8 Marks)**

Q. 47 Write a short note on domain hijacking. **(4 Marks)**

Q. 48 List the countermeasures against domain hijacking. **(6 Marks)**

Q. 49 Describe DNS Flood Attack and list its countermeasures. **(8 Marks)**

Q. 50 Describe Distributed Reflection Denial of Service (DRDoS) and list its countermeasures. **(8 Marks)**

Q. 51 Describe DNS Cache Poisoning. **(8 Marks)**

Q. 52 List the various countermeasures against DNS Cache Poisoning. **(6 Marks)**

Q. 53 Write a short note on DNS Tunnelling. **(4 Marks)**

Q. 54 Explain DNS Tunnelling and list its countermeasures. **(8 Marks)**

Q. 55 Describe Pseudo-Random Subdomain (PRSD) Attack and list its countermeasures. **(6 Marks)**

Q. 56 Describe NXDOMAIN Attack and list its countermeasures. **(6 Marks)**

Q. 57 Describe Phantom Domain Attack and list its countermeasures. **(6 Marks)**

Q. 58 Write a short note on Fast Flux. **(4 Marks)**

Email Attacks

Q. 59 Explain Business Email Compromise (BEC) and its variants. **(6 Marks)**

Q. 60 Write a short note on Malicious Attachments. **(4 Marks)**

Q. 61 Write a short note on Email Bombing. **(4 Marks)**

Q. 62 Describe Unsolicited Bulk Email (UBE) and explain UBE sending techniques. **(8 Marks)**

Q. 63 Write a short note on Unauthorised Email Senders. **(4 Marks)**

Q. 64 Write a short note on Unauthorised Email Receivers. **(4 Marks)**

Q. 65 Rosh ensures that he checks each email received every day and takes the intended action as soon as possible. Describe to him why he should not do that explaining that he could be a victim of Email Address Spoofing. **(8 Marks)**

Q. 66 List the scenarios that could potentially lead to Email Content Leakage. **(4 Marks)**

Q. 67 How is it possible to do tampering with email content? **(4 Marks)**

Q. 68 List common countermeasures against email attacks. **(6 Marks)**

Q. 69 What are some of the steps that you could take to protect email from malware? **(4 Marks)**

Q. 70 List some of the steps that you would take to for securing mail clients. **(6 Marks)**

Firewalls

Q. 71 What are the components of a firewall rule? **(4 Marks)**

Q. 72 Classify firewalls based on the OSI Layer. **(4 Marks)**

Q. 73 Classify firewalls based on Form Factor. **(4 Marks)**

Q. 74 Classify firewalls based on the type of Inspection. **(4 Marks)**

Q. 75 Classify firewalls based on architecture. **(8 Marks)**

Q. 76 What are some of the major challenges in managing and deploying firewalls? **(6 Marks)**

Penetration Testing

Q. 77 Describe penetration testing. **(4 Marks)**

Q. 78 How do you go about carrying out penetration testing. **(6 Marks)**

❑❑❑