Asset

An *asset* is an item of economic value owned by an organization or an individual. Identification of assets within the risk analysis world is the first and most important step. After all, if you don't know what you have, how can you possibly secure it? Assets can be anything from physical devices (such as desktops, servers, printers, switches, and routers) to databases and file shares.

Threat

A *threat* is any agent, circumstance, or situation that could cause harm or loss to an IT asset. Threats can take on many forms, and may not always be readily identifiable. For example, you probably already think of malicious hackers and viruses as threats, but what about bad weather? A hurricane, tornado, flood, or earthquake could cause just as much damage to your assets as a hacker could ever dream of doing. Ethical hackers are, obviously, much more concerned with the virtual threat agent techniques, but security professionals designing an entire program need to be cognizant of as many threats as possible.

Vulnerability

A *vulnerability* is any weakness, such as a software flaw or logic design, that could be exploited by a threat to cause damage to an asset. The goal of pen testers is to discover these vulnerabilities and attempt to exploit them. The key thing to remember about vulnerabilities is that their existence does not necessarily equate to a risk. For example, given physical access to any computer system, a hacker could easily (usually) successfully hack the device—so the vulnerability (physical access) exists. However, if your server is locked in an airtight room, buried in an underground silo, with multiple guards and physical security measures in place, the probability of it being exploited is reduced to near zero.

Confidentiality,Integrity and Availability

Some of the basics include the security triad of Confidentiality, Integrity, and Availability. Confidentiality, addressing the secrecy and privacy of information, refers to the measures taken to prevent disclosure of information or data to unauthorized individuals or systems. The use of passwords is by far the most common logical measure taken to ensure confidentiality, and attacks against passwords are, amazingly enough, the most common confidentiality attacks.

Integrity refers to the methods and actions taken to protect the information from unauthorized alteration or revision—whether the data is at rest or in transit. Integrity in information systems is often ensured through the use of a hash (a one-way mathematical algorithm such as MD5 or SHA-1).

Availability refers to the communications systems and data being ready for use when legitimate users need it. Denial of service (DoS) attacks are designed to prevent legitimate users from having access to a computer resource or service, and can take many forms.

Hackers and Types of Hackers

Hackers are generally classified into three separate groups. *White hats* are the ethical hackers hired by a customer for the specific goal of testing and improving security, or for other defensive purposes. *Black hats* are the crackers illegally using their skills for either personal gain or for malicious intent—and they do *not* ask for permission or consent. *Gray hats* are neither good nor bad—they are simply curious about hacking tools and techniques, or feel like it's their duty, with or without customer permission, to demonstrate security flaws in systems. In any case, hacking without a customer's explicit permission and direction is a crime.

What is Penetration testing?

A penetration test, also known as a pen test, is a clearly defined, full-scale test of the security controls of a system or network in order to identify security risks and vulnerabilities. The three main phases in a pen test are preparation, assessment, and conclusion. The preparation phase defines the time period when the actual contract is hammered out. The scope of the test, the types of attacks allowed, and the individuals assigned to perform the activity are all agreed upon in this phase. The *assessment* phase (sometimes also known as the *security evaluation* phase) is when the actual assaults on the security controls are conducted. The *conclusion* (or post-assessment) phase defines the time when final reports are prepared for the customer, detailing the findings of the test (including the types of tests performed) and many times even providing recommendations to improve security.

The act of hacking itself consists of five main phases. Reconnaissance involves the steps taken to gather evidence and information on the targets you wish to attack. It can be passive in nature or active. Scanning and enumeration takes the information gathered in recon and actively applies tools and techniques to gather more in-depth information on the targets. In the gaining access phase, true attacks are leveled against the targets enumerated in phase 2. In phase 4, maintaining access, hackers attempt to ensure they have a way back into the machine or system they've already compromised. Finally, in the final phase, covering tracks, attackers attempt to conceal their success and avoid detection by security professionals.

Attack Surface

An attack surface is the total sum of the vulnerabilities in a given computing device or network that are accessible to a hacker.

Anyone trying to break into a system generally starts by scanning the target's attack surface for possible attack vectors, whether for an active attack or passive attack, ethical hacking or a hacking competition.

Attack surfaces can be divided in to a few categories:

- The network attack surface.
- The software attack surface.
- The physical attack surface.

Every point of network interaction is a potential part of the network attack surface. A network attack surface can be reduced by closing unnecessarily open ports and limiting the resources that are available to untrusted users and to the Internet in general, through methods like MAC address filtering. Limiting network attack vectors can also limit the exposure of existing software vulnerabilities by blocking access to them.

As all running code has the possibility of having exploitable vulnerabilities, one of the first and simplest ways to limit software attack surface is to reduce the amount of running code. The more a piece of malware can use various exploits, the more chance it can get in via a hole in a target system's attack surface.

Physical access also constitutes an attack surface, which overlaps with the social engineering attack surface. This surface is exploitable by inside vectors such as rogue employees or hired workers. External risks include password retrieval from carelessly discarded hardware or from password sticky notes. Best practices for physical attack surface remediation include enforcing strong authentication, destroying hard drives before throwing them out and refraining from leaving hard copy access data -- like sticky note passwords – in proximity to a computer.

Knowledge of all elements of an organization's attack surface is crucial to proper setup of breach detection systems (BDS), firewalls, intrusion prevention systems, data policies and other security measures.

Malware

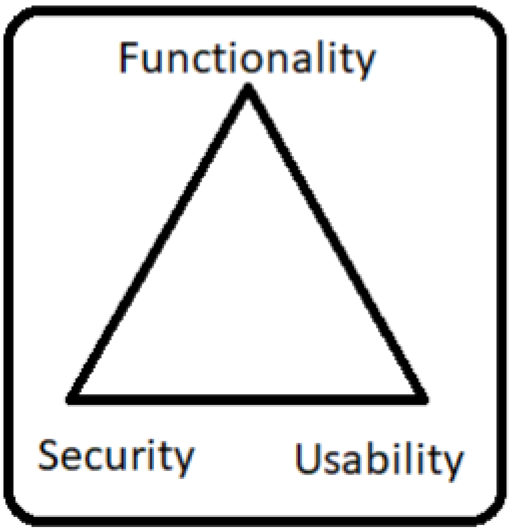
Malware, or malicious software, is any program or file that is harmful to a computer user. Malware includes computer viruses, worms, Trojan horses and spyware. These malicious programs can perform a variety of functions, including stealing, encrypting or deleting sensitive data, altering or hijacking core computing functions and monitoring users' computer activity without their permission.

Types of malware

There are different types of malware that contain unique traits and characteristics. A virus is the most common type of malware, and it's defined as a malicious program that can execute itself and spreads by infecting other programs or files. A worm is a type of malware that can self-replicate without a host program; worms typically spread without any human interaction or directives from the malware authors. A Trojan horse is a malicious program that is designed to appear as a legitimate program; once activated following installation, Trojans can execute their malicious functions. Spyware is a kind of malware that is designed to collect information and data on users and observe their activity without users' knowledge. A rootkit is a type of malware designed to obtain administrator-level access to the victim's system. Once installed, the program gives threat actors root or privileged access to the system.

Security, Functionality and Usability Triangle

There is an inter dependency between these three attributes. When security goes up, usability and functionality come down. Any organization should balance between these three qualities to arrive at a balanced information system.



In an ideal world, security professionals would like to have the highest level of security on all systems; however, sometimes this isn’t possible. Too many security barriers make it difficult for users to use the system and impede the system’s functionality. Suppose that in order to gain entry to your office at work, you had to first pass through a guard checkpoint at the entrance to the parking lot to verify your license plate number, then show a badge as you entered the building, then use a passcode to gain entry to the elevator, and finally use a key to unlock your office door. You might feel the security checks were too stringent! Any one of those checks could cause you to be detained and consequently miss an important meeting— for example, if your car was in the repair shop and you had a rental car, or you forgot your key or badge to access the building, elevator, or office door.

cross-site scripting (XSS)

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

cross site request forgery (CSRF/XSRF)

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated. CSRF attacks specifically target state-changing requests, not theft of data, since the attacker has no way to see the response to the forged request. With a little help of social engineering (such as sending a link via email or chat), an attacker may trick the users of a web application into executing actions of the attacker's choosing. If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.

SQL injection

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands.

input parameter manipulation

broken authentication

Authentication and session management includes all aspects of handling user authentication and managing active sessions. Authentication is a critical aspect of this process, but even solid authentication mechanisms can be undermined by flawed credential management functions, including password change, forgot my password, remember my password, account update, and other related functions. Because "walk by" attacks are likely for many web applications, all account management functions should require reauthentication even if the user has a valid session id.

User authentication on the web typically involves the use of a userid and password. Stronger methods of authentication are commercially available such as software and hardware based cryptographic tokens or biometrics, but such mechanisms are cost prohibitive for most web applications. A wide array of account and session management flaws can result in the compromise of user or system administration accounts. Development teams frequently underestimate the complexity of designing an authentication and session management scheme that adequately protects credentials in all aspects of the site. Web applications must establish sessions to keep track of the stream of requests from each user. HTTP does not provide this capability, so web applications must create it themselves. Frequently, the web application environment provides a session capability, but many developers prefer to create their own session tokens. In either case, if the session tokens are not properly protected, an attacker can hijack an active session and assume the identity of a user. Creating a scheme to create strong session tokens and protect them throughout their lifecycle has proven elusive for many developers. Unless all authentication credentials and session identifiers are protected with SSL at all times and protected against disclosure from other flaws, such as cross site scripting, an attacker can hijack a user's session and assume their identity.

sensitive information disclosure

XML External Entities

An *XML External Entity* attack is a type of attack against an application that parses XML input. This attack occurs when **XML input containing a reference to an external entity is processed by a weakly configured XML parser**. This attack may lead to the disclosure of confidential data, denial of service, server side request forgery, port scanning from the perspective of the machine where the parser is located, and other system impacts.

The XML 1.0 standard defines the structure of an XML document. The standard defines a concept called an entity, which is a storage unit of some type. There are a few different types of entities, external general/parameter parsed entity often shortened to **external entity**, that can access local or remote content via a declared system identifier. The system identifier is assumed to be a URI that can be dereferenced (accessed) by the XML processor when processing the entity. The XML processor then replaces occurrences of the named external entity with the contents dereferenced by the system identifier. If the system identifier contains tainted data and the XML processor dereferences this tainted data, the XML processor may disclose confidential information normally not accessible by the application. Similar attack vectors apply the usage of external DTDs, external stylesheets, external schemas, etc. which, when included, allow similar external resource inclusion style attacks.

Attacks can include disclosing local files, which may contain sensitive data such as passwords or private user data, using file: schemes or relative paths in the system identifier. Since the attack occurs relative to the application processing the XML document, an attacker may use this trusted application to pivot to other internal systems, possibly disclosing other internal content via http(s) requests or launching a CSRF attack to any unprotected internal services. In some situations, an XML processor library that is vulnerable to client-side memory corruption issues may be exploited by dereferencing a malicious URI, possibly allowing arbitrary code execution under the application account. Other attacks can access local resources that may not stop returning data, possibly impacting application availability if too many threads or processes are not released.

Note that the application does not need to explicitly return the response to the attacker for it to be vulnerable to information disclosures. An attacker can leverage DNS information to exfiltrate data through subdomain names to a DNS server that he/she controls.

Broken access control

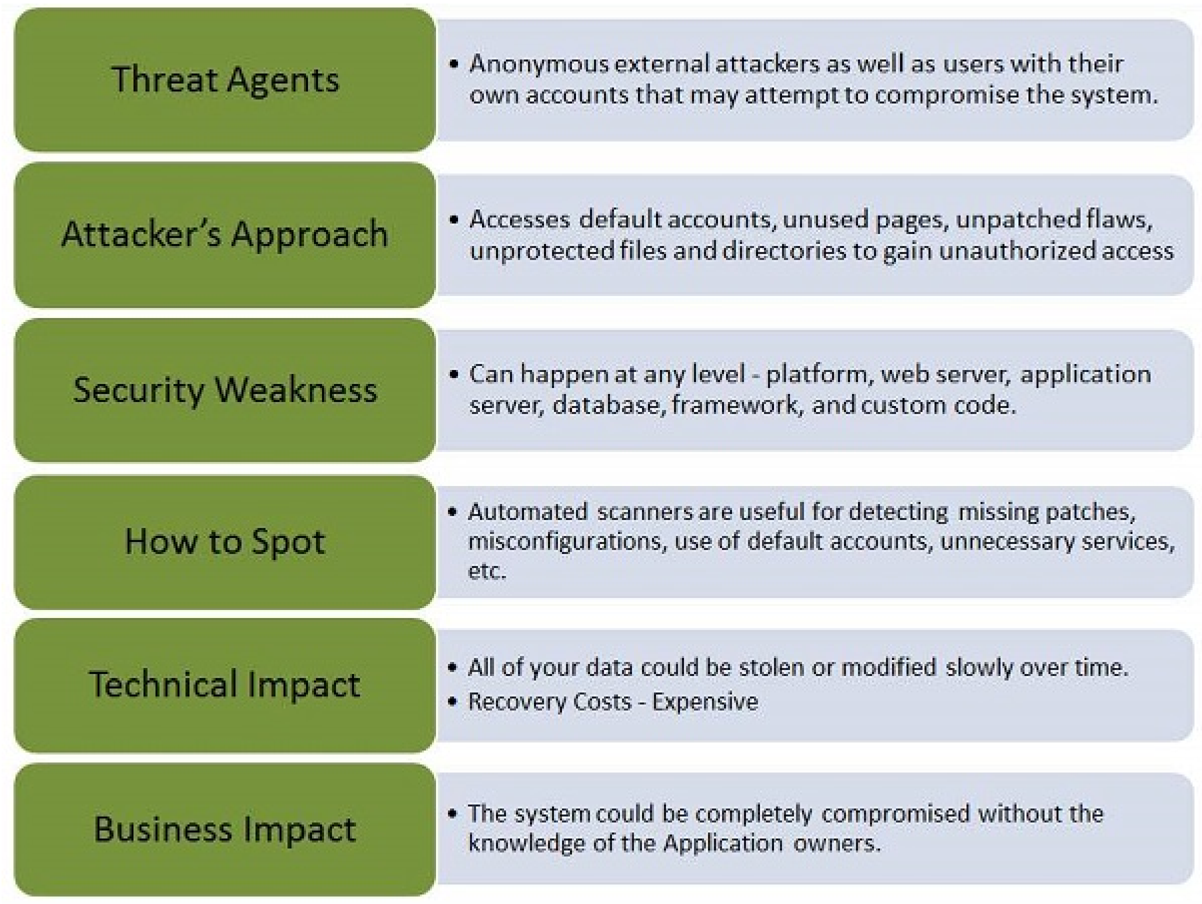
Access control, sometimes called authorization, is how a web application grants access to content and functions to some users and not others. These checks are performed after authentication, and govern what 'authorized' users are allowed to do.

Developers frequently underestimate the difficulty of implementing a reliable access control mechanism. Many of these schemes were not deliberately designed, but have simply evolved along with the web site. In these cases, access control rules are inserted in various locations all over the code. As the site nears deployment, the ad hoc collection of rules becomes so unwieldy that it is almost impossible to understand.

Many of these flawed access control schemes are not difficult to discover and exploit. Frequently, all that is required is to craft a request for functions or content that should not be granted. Once a flaw is discovered, the consequences of a flawed access control scheme can be devastating. In addition to viewing unauthorized content, an attacker might be able to change or delete content, perform unauthorized functions, or even take over site administration.

Security Misconfiguration

Security Misconfiguration arises when Security settings are defined, implemented, and maintained as defaults. Good security requires a secure configuration defined and deployed for the application, web server, database server, and platform. It is equally important to have the software up to date.



Threat Agents	• Anonymous external attackers as well as users with their own accounts that may attempt to compromise the system.
Attacker's Approach	• Accesses default accounts, unused pages, unpatched flaws, unprotected files and directories to gain unauthorized access
Security Weakness	• Can happen at any level - platform, web server, application server, database, framework, and custom code.
How to Spot	• Automated scanners are useful for detecting missing patches, misconfigurations, use of default accounts, unnecessary services, etc.
Technical Impact	• All of your data could be stolen or modified slowly over time. • Recovery Costs - Expensive
Business Impact	• The system could be completely compromised without the knowledge of the Application owners.

Keystroke Logging

A keylogger (keystroke logging) is a type of surveillance software that once installed on a system, has the capability to record every keystrokemade on that system. The Recording is saved in a log file, usually encrypted.

A keylogger can record instant messages, email, and capture any information you type at any time using your keyboard, including usernames, passwords and other personally identifiable information(pii). The log file created by the keylogger can then be sent to a specified receiver. Some keylogger programs will also record any email addresses you use and the URLs of any websites you visit.

Who Uses Keyloggers?

Keyloggers, as a surveillance tool, are often used by employers to ensure employees use work computers for business purposes only. There's also a growing market of parents who want to use keyloggers to stay informed about a child's online activities.

Denial of Service (DoS /DDoS)

A **Denial-of-Service (DoS) attack** is an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash. In both instances, the DoS attack deprives legitimate users (i.e. employees, members, or account holders) of the service or resource they expected.

Victims of DoS attacks often target web servers of high-profile organizations such as banking, commerce, and media companies, or government and trade organizations. Though DoS attacks do not typically result in the theft or loss of significant information or other assets, they can cost the victim a great deal of time and money to handle.

There are two general methods of DoS attacks: flooding services or crashing services. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop. Popular flood attacks include:

- **Buffer overflow attacks** – the most common DoS attack. The concept is to send more traffic to a network address than the programmers have built the system to handle. It includes the attacks listed below, in addition to others that are designed to exploit bugs specific to certain applications or networks
- **ICMP flood** – leverages misconfigured network devices by sending spoofed packets that ping every computer on the targeted network, instead of just one specific machine. The network is then triggered to amplify the traffic. This attack is also known as the smurf attack or ping of death.
- **SYN flood** – sends a request to connect to a server, but never completes the handshake. Continues until all open ports are saturated with requests and none are available for legitimate users to connect to.

Waterhole attack

A watering hole attack is a security exploit in which the attacker seeks to compromise a specific group of end users by infecting websites that members of the group are known to visit. The goal is to infect a targeted user's computer and gain access to the network at the target's place of employment.Watering hole attack is inspired by predators in the natural world who lurk near watering holes, looking for opportunities to attack desired prey. In a watering hole attack, the predator lurks near niche websites popular with the target prey, looking for opportunities to infect the websites with malware or malvertisementsthat will make the target vulnerable.

Watering hole attacks, which tend to focus on legitimate, popular websites, are a derivative of pivot attacks, which target one thing to get at another. In a watering hole attack, the attacker first profiles its targets -- who are typically employees of large enterprises, human rights groups or government offices -- to determine the type of websites they frequent. The attacker then looks for vulnerabilities in the websites and injects malicious JavaScript or HTML code that redirects the target to a separate site where the malware is hosted. This compromised website is now ready to infect the target with the injected malware upon access.

While watering hole attacks are uncommon, they pose a considerable threat since they are difficult to detect and typically target high-security organizations through their low-security employees, business partners, connected vendors or an unsecured wireless network.

Brute Force

A **Brute Force Attack** is the simplest method to gain access to a site or server (or anything that is password protected). It tries various combinations of usernames and passwords again and again until it gets in. This repetitive action is like an army attacking a fort.

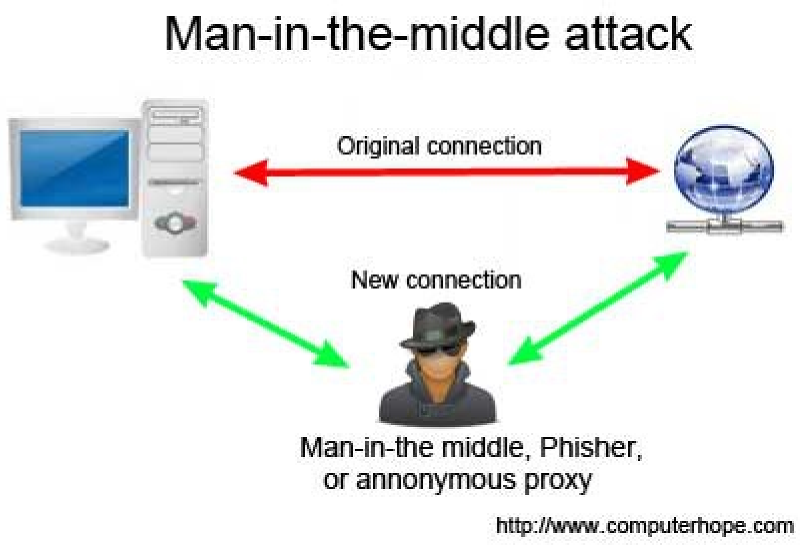
Phishing and Fake WAP

Eavesdropping

An eavesdropping attack, which are also known as a sniffing or snooping attack, is an incursion where someone tries to steal information that computers, smartphones, or other devices transmit over a network. An eavesdropping attack takes advantage of unsecured network communications in order to access the data being sent and received. Eavesdropping attacks are difficult to detect because they do not cause network transmissions to appear to be operating abnormally.

Man-in-the-middle

The Man-in-the-Middle attack (abbreviated MITM, MitM, MIM, MiM, MITMA) implies an active attack where the adversary impersonates the user by creating a connection between the victims and sends messages between them. In this case, the victims think that they are communicating with each other, but in reality, the malicious actor controls the communication.


Man-in-the-middle attack
Original connection
New connection
Man-in-the middle, Phisher, or annonymous proxy
http://www.computerhope.com

A third person exists to control and monitor the traffic of communication between two parties. Some protocols such as **SSL** serve to prevent this type of attack.

Session Hijacking

The Session Hijacking attack consists of the exploitation of the web session control mechanism, which is normally managed for a session token.

Because http communication uses many different TCP connections, the web server needs a method to recognize every user's connections. The most useful method depends on a token that the Web Server sends to the client browser after a successful client authentication. A session token is normally composed of a string of variable width and it could be used in different ways, like in the URL, in the header of the http requisition as a cookie, in other parts of the header of the http request, or yet in the body of the http requisition.

The Session Hijacking attack compromises the session token by stealing or predicting a valid session token to gain unauthorized access to the Web Server.

The session token could be compromised in different ways; the most common are:

- Predictable session token;
- Session Sniffing;
- Client-side attacks (XSS, malicious JavaScript Codes, Trojans, etc);
- Man-in-the-middle attack
- Man-in-the-browser attack

Clickjacking

Clickjacking, also known as a "UI redress attack", is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the the top level page. Thus, the attacker is "hijacking" clicks meant for their page and routing them to another page, most likely owned by another application, domain, or both.

Using a similar technique, keystrokes can also be hijacked. With a carefully crafted combination of stylesheets, iframes, and text boxes, a user can be led to believe they are typing in the password to their email or bank account, but are instead typing into an invisible frame controlled by the attacker.

Cookie Theft

Browser cookies are very visible and can easily stolen or manipulated.

Some web browsers show all cookie data by looking in the preferences area. Lately, it has become more commonplace for browsers to hide this information, but that does not mean that cookie storage is less visible to an attacker. Stored cookies can also be stolen using Cross-Site Scripting (XSS).

Cookie data is also visible while in transit. It is sent in plain text in the headers of every request to the webserver and can be seen by an attacker who can observer network traffic. This is especially easy to do on an open WiFi network such as those commonly found at coffee shops and other businesses.

URL Obfuscation

An obfuscated URL is a web address that has been obscured or concealed and has been made to imitate the original URL of a legitimate website. It is done to make users access a spoof website rather than the intended destination.

Obfuscated URLs are one of the many phishing attacks that can fool Internet users. The spoof site is often an identical clone of the original one in order to fool users into divulging login and other personal information.

An obfuscated URL is also called a hyperlink trick.

Attackers usually use a common misspelling technique where they misspell a domain name to trick users into visiting. These obfuscated URLs can be a cause of malware entering a user’s computer system.

URL obfuscation is used together with spamming, redirecting users using a misleading URL that leads to a malicious site. URLs are strings of text that identify web resources such as websites or any kind of Internet server, so an obfuscated URL shows up as a meaningless query string to users.

This hides the real address of the linked site when the user hovers over the link. URL obfuscation is not always used for phishing or cross-site scripting, but it is also used by legitimate websites to hide the true URLs of certain pages so that they cannot be accessed directly by the users or allow certain procedures to be bypassed. It is also used as an anti-hacking procedure. This is termed as security through obscurity.

DNS poisoning

DNS Poisoning is a technique that tricks a DNS server into believing that it has received authentic information when, in reality, it has not. It results in the substitution of false IP address at the DNS level where web addresses are converted into numeric IP addresses. It allows an attacker to replace IP address entries for a target site on a given DNS server with IP address of the server controls. An attacker can create fake DNS entries for the server which may contain malicious content with the same name.

For instance, a user types www.google.com, but the user is sent to another fraud site instead of being directed to Google's servers. As we understand, DNS poisoning is used to redirect the users to fake pages which are managed by the attackers.

ARP poisoning

Address Resolution Protocol (ARP) is a stateless protocol used for resolving IP addresses to machine MAC addresses. All network devices that need to communicate on the network broadcast ARP queries in the system to find out other machines' MAC addresses. ARP Poisoning is also known as **ARP Spoofing**.

Here is how ARP works −

- When one machine needs to communicate with another, it looks up its ARP table.
- If the MAC address is not found in the table, the **ARP_request** is broadcasted over the network.
- All machines on the network will compare this IP address to MAC address.
- If one of the machines in the network identifies this address, then it will respond to the **ARP_request** with its IP and MAC address.
- The requesting computer will store the address pair in its ARP table and communication will take place.

What is ARP Spoofing?

ARP packets can be forged to send data to the attacker's machine.

- ARP spoofing constructs a large number of forged ARP request and reply packets to overload the switch.
- The switch is set in **forwarding mode** and after the **ARP table** is flooded with spoofed ARP responses, the attackers can sniff all network packets.

Attackers flood a target computer ARP cache with forged entries, which is also known as **poisoning**. ARP poisoning uses Man-in-the-Middle access to poison the network.

Identity Theft

Identity theft, also known as identity fraud, is a crime in which an imposter obtains key pieces of personally identifiable information, such as Social Security or driver's license numbers, in order to impersonate someone else.

The information can be used to obtain credit, merchandise and services in the name of the victim, or to provide the thief with false credentials. In addition to running up debt, in rare cases, an imposter might provide false identification to police, creating a criminal record or leaving outstanding arrest warrants for the person whose identity has been stolen.

Types and examples of identity theft

Identity theft is categorized two ways: true name and account takeover. True-name identity theft means the thief uses personal information to open new accounts. The thief might open a new credit card account, establish cellular phone service or open a new checking account in order to obtain blank checks.

Account-takeover identity theft means the imposter uses personal information to gain access to the person's existing accounts. Typically, the thief will change the mailing address on an account and run up a huge bill before the person whose identity has been stolen realizes there is a problem. The internet has made it easier for an identity thief to use the information they've stolen, because transactions can be made without any personal interaction.

IoT Attacks

The IoT attack surface is the sum total of all potential security vulnerabilities in IoT devices and associated software and infrastructure in a given network, be it local or the entire Internet.

A thing, in the Internet of Things, can be any natural or man-made object that can be assigned an IP address and provided with the ability to transfer data over a network. A recent study from Hewlett Packard concluded that 70 percent of IoT devices contain serious vulnerabilities.

Hackers and government agencies can use vulnerabilities in IoT devices to gain access to a network to monitor users and potentially gain access to any other connected devices for any number of purposes. According to many security experts, our dependence on Internet-connected technology is outpacing our ability to secure it. Joshua Corman, a security strategist and the chief technology officer at the software firm Sonatype, explains:

“You’re taking things that weren’t connected and weren’t vulnerable and putting vulnerability and connectivity on all of them. So if the Internet is a perfect surveillance machine, what happens with the Internet of Things? It’s just gonna take that to the next order of magnitude.”

BOTs and BOTNETs

When computers get infected with malware bots, they could be included to a network of infected computers, forming botnets. These botnets will be orchestrated by a command and control center that instructs them on specific malicious actions. A computer infected with a malware bot or virus can spread the same in their intranet, creating massive botnets. In most cases, the users of these computers are not aware that theirs is a part of a botnet, performing malicious activities.

Botnets are created to perform malicious activities such as Distributed Denial of Service (DDoS) attacks, phishing scams, spam emails, ransomware, click fraud and a lot more.

In most cases, computers become infected and turn into botnets because of a weak end-point security system. This can be taken care of by having the virus and malware programs and definitions updated and patched. Also, users of these computers should be educated on the perils of opening unknown attachments and clicking on suspicious executables.

Bots are computer programs or software applications designed to execute a series of operations automatically. There are several useful bots (good bots) that crawl websites and create visibility on search engines and social media channels. There are several bots executed for general information collection, like weather reports over several locations across the globe, football scores and team performances over time, and so on. Bots are employed for these activities because they're either mundane or too repetitive for humans to perform.

There are bots (bad bots) created to cause harm to websites and online businesses. Bots are created to illegally scrape content from websites and post them elsewhere. Competitors employ third-party scrapers to gather information and content, so that they can fine tune their business strategies to overtake the competition, unethically. Millions of bots sent from single or multiple IPs can cause Denial of Service (DoS) attacks, stifle bandwidth and severly impact user experience.

Having said that, the hackers creating bots, targeting online businesses, seem to be moving to sophisticated methods to get around the security mechanism in place. Real-time bot prevention solutions that constantly learn and update bot signatures and patterns will provide continuous protection to websites.