Cryptography and System Security (CSS)

Course Code: CSC 604

Subject Incharge

Ankita Karia

Assistant Professor

Room No. 421

email: ankitakaria@sfit.ac.in

Module 6: System Security

6.1

- Software Vulnerabilities
- Buffer Overflow, Format string, cross-site scripting,
- SQL injection,
- Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits

Software Vulnerability

1. A software vulnerability is a glitch, flaw, or weakness present in the software or in an OS (Operating System).
2. The severity of software vulnerabilities advances at an exponential rate.
3. Of course, all systems include vulnerabilities. The thing is whether or not they're exploited to cause damage.

Software vulnerabilities are explained by three ideal factors. These are:

Existence – The existence of a vulnerability in the software.

Access – The possibility that hackers gain access to the vulnerability.

Exploit – The capability of the hacker to take advantage of that vulnerability via tools or with certain techniques.

Most Common Software Vulnerabilities

According to the OWASP Top 10, here are the most common vulnerabilities:

1. Insufficient Logging and Monitoring

Insufficient logging and monitoring processes are dangerous. This leaves your data vulnerable to tampering, extraction, or even destruction.

2. Injection Flaws

Injection flaws occur when untrusted data is sent as part of a command or query. The attack can then trick the targeted system into executing unintended commands. An attack can also provide untrustworthy agents access to protected data.

3. Sensitive Data Exposure

Sensitive data — such as addresses, passwords, and account numbers — must be properly protected. If it isn't, untrustworthy agents take advantage of the vulnerabilities to gain access.

Most Common Software Vulnerabilities

4. Using Components with Known Vulnerabilities

Components are made up of libraries, frameworks, and other software modules. Often, the components run on the same privileges as your application. If a component is vulnerable, it can be exploited by an untrustworthy agent. This causes serious data loss or server takeover.

5. Cross-Site Scripting (XSS) Flaws

Untrustworthy agents can take advantage of cross-site scripting flaws to execute their own scripts in the targeted system. In general, cross-site scripting flaws happen in one-of-two ways:

a. Whenever an application includes untrusted data in a new web page without proper validation.

b. Whenever an existing webpage is updated with user-supplied data using a browser API that can create HTML or JavaScript.

Most Common Software Vulnerabilitiesz

6. Broken Authentication

Authentication and session management application functions need to be implemented correctly. If they aren't, it creates a software vulnerability that can be exploited by untrustworthy agents to gain access to personal information.

7. Broken Access Control

User restrictions must be properly enforced. If they are broken, it can create a software vulnerability. Untrustworthy agents can exploit that vulnerability.

8. XML External Entities (XXE)

XML is a popular data format that is used in web services, documents, and image files. You need an XML parser to understand XML data. But if it's poorly configured and the XML input that contains a reference to an external entity, it's dangerous. An untrustworthy agent can cause a DoS.

Most Common Software Vulnerabilities

9. Security Misconfiguration: Security misconfigurations are often the result of:

a. Insecure default configurations.

b. Incomplete or impromptu configurations.

c. Open Cloud storage.

d. Misconfigured HTTP headers.

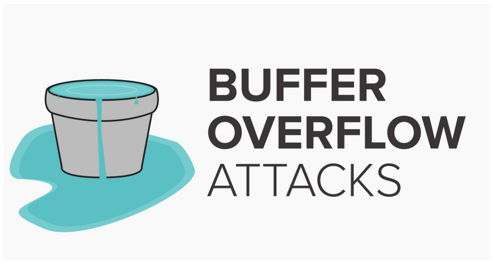
e. Wordy error messages that contain sensitive information.

10. Insecure Deserialization

Deserialization flaws often result in remote code execution. This enables untrustworthy agents to perform replay, injection, and privilege escalation attacks.

Buffer Overflow Attack – WHAT IS BUFFER

1. A buffer, or data buffer, is an area of physical memory storage used to temporarily store data while it is being moved from one place to another. These buffers typically live in RAM memory.
2. Computers frequently use buffers to help improve performance; most modern hard drives take advantage of buffering to efficiently access data, and many online services also use buffers.
3. For example, buffers are frequently used in online video streaming to prevent interruption. When a video is streamed, the video player downloads and stores perhaps 20% of the video at a time in a buffer and then streams from that buffer. This way, minor drops in connection speed or quick service disruptions won't affect the video stream performance.



Buffer Overflow Attack – WHAT IS BUFFER

1. Buffers are designed to contain specific amounts of data. Unless the program utilizing the buffer has built-in instructions to discard data when too much is sent to the buffer, the program will overwrite data in memory adjacent to the buffer.

2. Buffer overflows can be exploited by attackers to corrupt software.

3. Despite being well-understood, buffer overflow attacks are still a major security problem that torment cyber-security teams.

4. In 2014 a threat known as ‘heartbleed’ exposed hundreds of millions of users to attack because of a buffer overflow vulnerability in SSL software.

Buffer Overflow Attack – How do attackers exploit buffer overflows?

1. An attacker can deliberately feed a carefully crafted input into a program that will cause the program to try and store that input in a buffer that isn’t large enough, overwriting portions of memory connected to the buffer space.
2. If the memory layout of the program is well-defined, the attacker can deliberately overwrite areas known to contain executable code.
3. The attacker can then replace this code with his own executable code, which can drastically change how the program is intended to work.

For example if the overwritten part in memory contains a pointer (an object that points to another place in memory) the attacker’s code could replace that code with another pointer that points to an exploit payload. This can transfer control of the whole program over to the attacker’s code.

Buffer Overflow Attack – Who is vulnerable to buffer overflow attacks?

1. Certain coding languages are more susceptible to buffer overflow than others.
2. C and C++ are two popular languages with high vulnerability, since they contain no built-in protections against accessing or overwriting data in their memory.
3. Windows, Mac OSX, and Linux all contain code written in one or both of these languages.

More modern languages like Java, PERL, and C# have built-in features that help reduce the chances of buffer overflow, but cannot prevent it altogether.

Format String Attack

1. A Format String attack can occur when an input string's submitted data is evaluated as a command by the application.

2. Taking advantage of a Format String vulnerability, an attacker can execute code, read the Stack, or cause a segmentation fault in the running application – causing new behavior that compromise the security or the stability of the system.

3. Format String attacks alter the flow of an application.

4. They use string formatting library features to access other memory space. Vulnerabilities occurred when the user-supplied data is deployed directly as formatting string input for certain C/C++ functions

Format String Attack

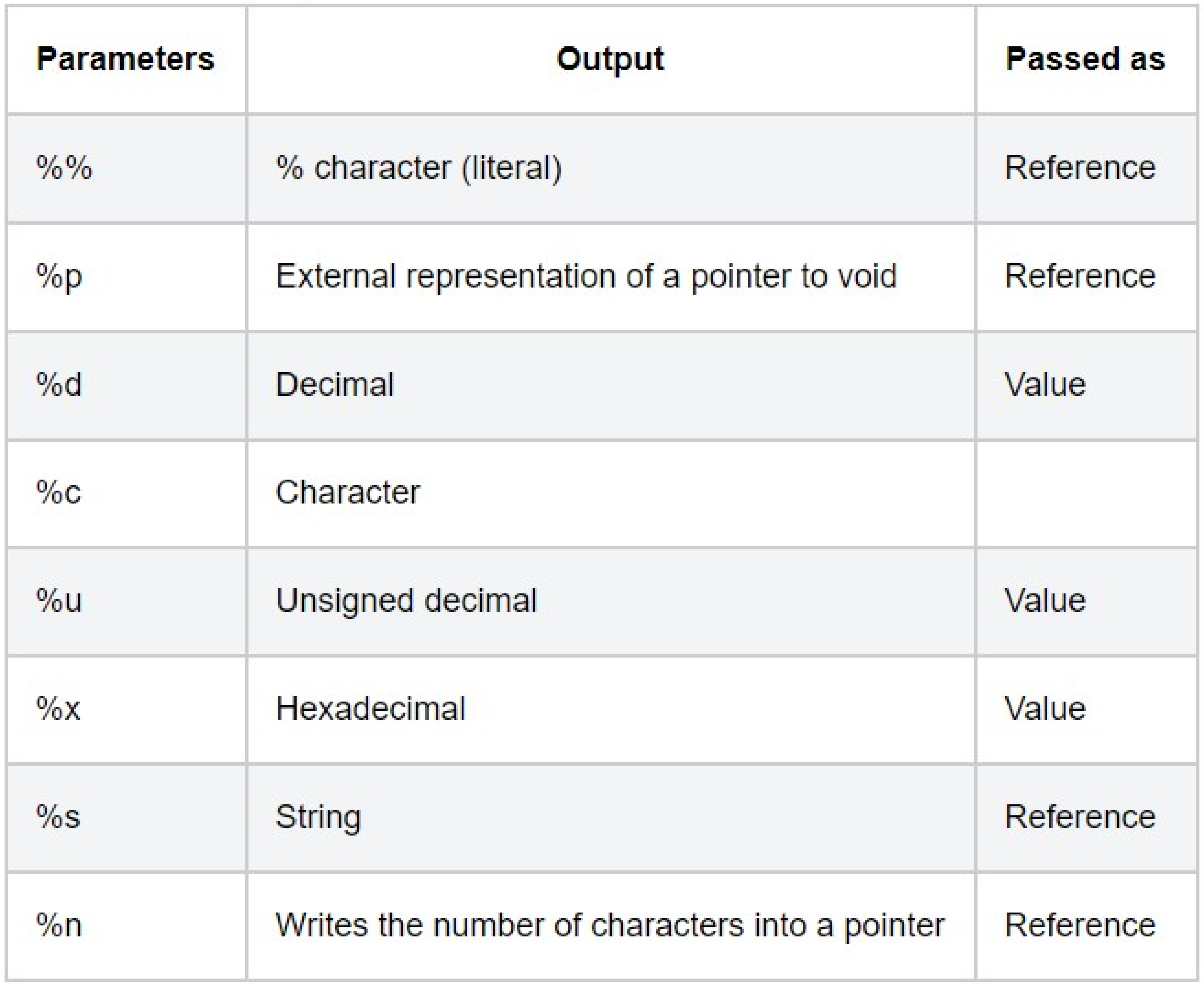
To understand the attack, it's necessary to understand the components that constitute it.

1. The **Format Function** is an ANSI C conversion function, like printf, fprintf, which converts a primitive variable of the programming language into a human-readable string representation.
2. The **Format String is the argument of the Format Function** and is an ASCII string which contains text and format parameters, like: **printf ("The magic number is: %d\n", 1911);**
3. The **Format String Parameter, like %x %s** defines the type of conversion of the format function.

The attack could be executed when the application doesn't properly validate the submitted input. In this case, if a Format String parameter, like %x, is inserted into the posted data, the string is parsed by the Format Function, and the conversion specified in the parameters is executed. However, the Format Function is expecting more arguments as input, and if these arguments are not supplied, the function could read or write the stack.

Format String Attack

Table 2. Common parameters used in a Format String Attack.



Parameters	Output	Passed as
%%	% character (literal)	Reference
%p	External representation of a pointer to void	Reference
%d	Decimal	Value
%c	Character	
%u	Unsigned decimal	Value
%x	Hexadecimal	Value
%s	String	Reference
%n	Writes the number of characters into a pointer	Reference

Format String Attack

```
#include <stdio.h>
void main(int argc, char **argv)
    {
              // This line is safe
               printf("%s\n", argv[1]);
```

→ **SAFE CODE**

The line printf("%s", argv[1]); in the example is safe, if you compile the program and run it as follows:

```
./example "Hello World %s%s%s%s%s%s“
```

The printf in the first line will not interpret the “%s%s%s%s%s%s” in the input string,

and the output will be: “Hello World %s%s%s%s%s%s”

Format String Attack

```
#include <stdio.h>
void main(int argc, char **argv)
    {
            // This line is vulnerable
            printf(argv[1]);
    }
```

→ **VULNERABLE CODE**

The printf in the above line will interpret the %s%s%s%s%s%s in the input string as a reference to string pointers, so it will try to interpret every %s as a pointer to a string, starting from the location of the buffer (probably on the Stack).

At some point, it will get to an invalid address, and attempting to access it will cause the program to crash.

Format String Attack

An attacker can also use this to get information, not just crash the software.
For example, running:

```
./example "Hello World %p %p %p %p %p %p"
```

Will print the lines:

```
Hello World %p %p %p %p %p %p
```

→ **Output from safe code**

```
Hello World 000E133E 000E133E 0057F000 CCCCCCCC CCCCCCCC CCCCCCCC
```

↑ **Output from vulnerable code**

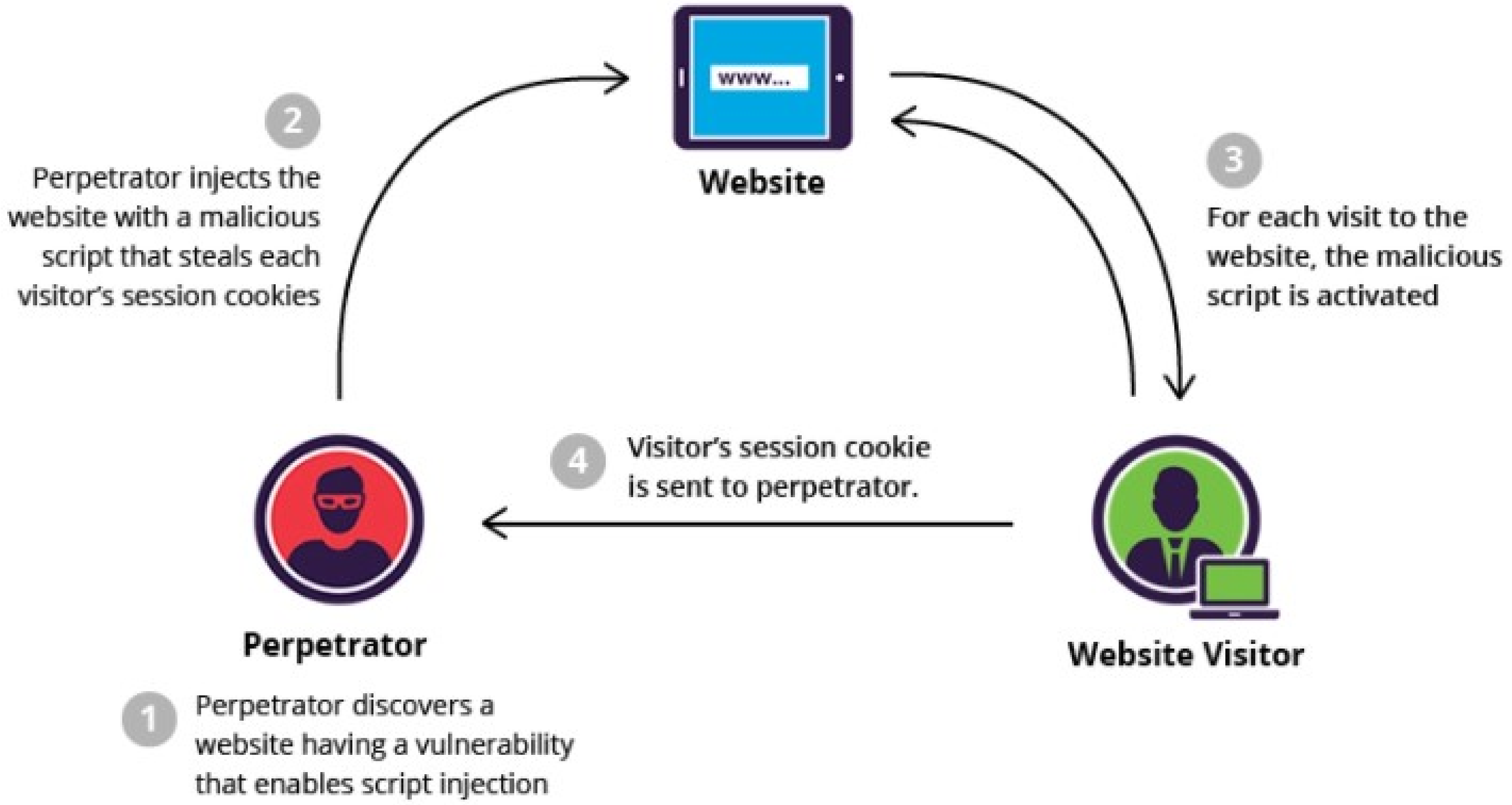
The values printed after the “Hello World” text, are the values on the stack of computer at the moment of running this example.

Also reading and writing to any memory location is possible in some conditions, and even code execution.

Cross Site Scripting Attack (XSS Attack)

1. Cross site scripting (XSS) is a common attack vector that injects malicious code into a vulnerable web application.
2. XSS differs from other web attack vectors (e.g., SQL injections), in that it does not directly target the application itself. Instead, the users of the web application are the ones at risk.
3. Cross-site Scripting (XSS) is a client-side code injection attack.
4. The attacker aims to execute malicious scripts in a web browser of the victim by including malicious code in a legitimate web page or web application.
5. The actual attack occurs when the victim visits the web page or web application that executes the malicious code. The web page or web application becomes a vehicle to deliver the malicious script to the user's browser.
6. Vulnerable vehicles that are commonly used for Cross-site Scripting attacks are forums, message boards, and web pages that allow comments.

Cross Site Scripting Attack (XSS Attack) – How does it work


www...
Website
2
Perpetrator injects the website with a malicious script that steals each visitor's session cookies
3
For each visit to the website, the malicious script is activated
4
Visitor's session cookie is sent to perpetrator.
Perpetrator
Website Visitor
1
Perpetrator discovers a website having a vulnerability that enables script injection

Cross Site Scripting Attack (XSS Attack) – How does it work

1. To successfully execute a stored XSS attack, a perpetrator has to locate a vulnerability in a web application and then inject malicious script into its server (e.g., via a comment field).
2. When attackers inject their own code into a web page, typically accomplished by exploiting a vulnerability on the website’s software, they can then inject their own script, which is executed by the victim’s browser.
3. Since the JavaScript runs on the victim’s browser page, sensitive details about the authenticated user can be stolen from the session, essentially allowing a bad actor to target site administrators and completely compromise a website.
4. One of the most frequent targets are websites that allow users to share content, including blogs, social networks, video sharing platforms and message boards.
5. Every time the infected page is viewed, the malicious script is transmitted to the victim’s browser.

Cross Site Scripting Attack (XSS Attack) – Types

Depending on their goals, bad actors can use cross-site scripting in a number of different ways.

Some of the most common types of attacks.

1. **Stored (Persistent) Cross Site-Scripting**
2. **Reflected Cross-Site Scripting**
3. **Self Cross-Site Scripting**
4. **Blind Cross-Site Scripting**
5. **DOM-Based Cross-Site Scripting**

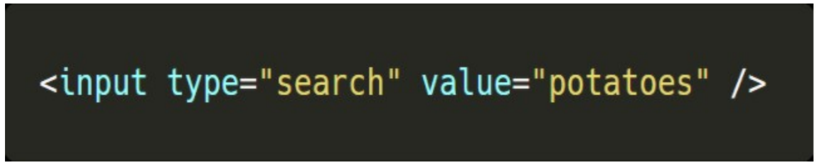
Cross Site Scripting Attack (XSS Attack) – Types

Stored (Persistent) Cross Site-Scripting: Occurs when attackers stores their payload on a compromised server, causing the website to deliver malicious code to other visitors. This method only requires an initial action from the attacker and can compromise many visitors afterwards, this is the most dangerous and most commonly employed type of cross-site scripting. Examples of stored cross-site scripting attacks include the profile fields such as your username or email, which are saved on the server and displayed on your account page.

Cross Site Scripting Attack (XSS Attack) – Types

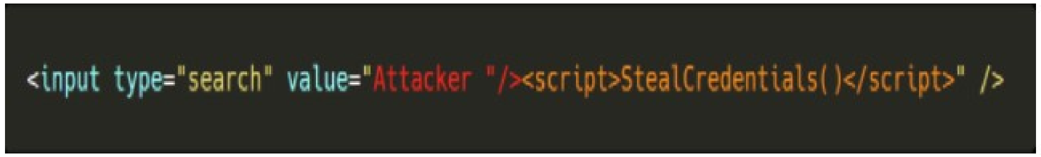
Reflected Cross-Site Scripting:

- Occurs when the payload is stored in the data sent from the browser to the server.
- Examples of reflected cross-site scripting attacks include when an attacker stores malicious script in the data sent from a **website's search or contact form.**
- A typical example of reflected cross-site scripting is a search form, where visitors sends their search query to the server, and only they see the result.



```
<input type="search" value="potatoes" />
```

- Attackers typically send victims custom links that direct unsuspecting users toward a vulnerable page. From this page, they often employ a variety of methods to trigger their proof of concept



```
<input type="search" value="Attacker "/><script>StealCredentials()</script>" />
```

Cross Site Scripting Attack (XSS Attack) – Types

Self Cross-Site Scripting: Self cross-site scripting occurs when attackers exploit a vulnerability that requires extremely specific context and manual changes. The only one who can be a victim is yourself. These specific changes can include things like cookie values or setting your own information to a payload.

Blind Cross-Site Scripting: Blind cross-site scripting attacks occur when an attacker can't see the result of an attack. In these attacks, the vulnerability commonly lies on a page where only authorized users can access. This method requires more preparation to successfully launch an attack; if the payload fails, the attacker won't be notified.

Cross Site Scripting Attack (XSS Attack) – Types

DOM-Based Cross-Site Scripting: Occurs when the server itself isn’t the one vulnerable to XSS, but rather the JavaScript on the page is. As JavaScript is used to add interactivity to the page, arguments in the URL can be used to modify the page after it has been loaded. By modifying the DOM when it doesn’t sanitize the values derived from the user, attackers can add malicious code to a page. An example of DOM-based cross-site scripting attack would be when the website changes the language selection from the default one to one provided in the URL

SQL Injection Attack

1. SQL Injection (SQLi) is a type of an injection attack that allows an attacker to interfere with the queries that an application makes to its database.

2. It generally allows an attacker to view data that they are not normally able to retrieve. This might include data belonging to other users, or any other data that the application itself is able to access.

3. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior.

SQL Injection Attack

1. An SQL Injection vulnerability may affect any website or web application that uses an SQL database such as MySQL, Oracle, SQL Server, or others.
2. Criminals may use it to gain unauthorized access to your sensitive data: customer information, personal data, trade secrets, intellectual property, and more.
3. SQL Injection attacks are one of the oldest, most prevalent, and most dangerous web application vulnerabilities.
4. The OWASP organization (Open Web Application Security Project) lists injections in their OWASP Top 10 2017 document as the number one threat to web application security.

Malware: VIRUSES, Worms, Trojans, Logic Bomb, Bots, Rootkits

1. A virus is a specific type of malware that self-replicates by inserting its code into other programs.
2. Computer viruses have been prominent since almost the beginning of the commercial internet: The first one was created in 1982 for the Apple II, and other versions quickly followed.
3. Viruses spread by attaching themselves to legitimate files and programs, and are distributed through infected websites, flash drives, emails, software installations or unsecured links..
4. A victim activates a virus by opening the infected application or file.
5. Once activated, a virus may delete or encrypt files, modify applications, or disable system functions.

Malware: VIRUSES, Worms, Trojans, Logic Bomb, Bots, Rootkits

1. A computer **Virus is more dangerous than a computer worm** as it makes changes or deletes your files while worms only replicates itself with out making changes to your files/data.
2. Typical signs of computer virus infections include:
   a. Ongoing crashes and blue screen errors
   b. Slow performance
   c. Missing files
   d. Low storage
   e. Unexpected behavior
   f. Constant browser pop-ups
   g. Unidentifiable programs
   h. Increased network activity
   i. Disabled security software

Malware: VIRUSES, Worms, Trojans, Logic Bomb, Bots, Rootkits

File-infecting Virus A virus that attached itself to an executable program. It is **also called a parasitic virus** which typically infects files with .exe or .com extensions. Some file infectors can overwrite host files and others can damage your hard drive's formatting.	**Browser Hijacker** This virus targets and alters your browser setting. It is often called a browser redirect virus because it redirects your browser to other malicious websites that you don't have any intention of visiting. This virus can pose other threats such as changing the default home page of your browser.
Macro Virus This type of virus is commonly found in programs such as Microsoft Word or Excel. These viruses are usually stored as part of a document and can spread when the files are transmitted to other computers, often through email attachments.	**Web Scripting Virus** A very sneaky virus that targets popular websites. What this virus does is overwrite code on a website and insert links that can install malicious software on your device. Web scripting viruses can steal your cookies and use the information to post on your behalf on the infected website.

Malware: VIRUSES, Worms, Trojans, Logic Bomb, Bots, Rootkits

Boot Sector Virus These viruses are once common back when computers are booted from floppy disks. Today, these viruses are found distributed in forms of physical media such as external hard drives or USB. If the computer is infected with a boot sector virus, it automatically loads into the memory enabling control of your computer.	**Resident Virus** A resident virus stores itself on your computer's memory which allows it to infect files on your computer. This virus can interfere with your operating system leading to file and program corruption.
Polymorphic Virus This virus has the capability to evade anti-virus programs since it can change codes every time an infected file is performed.	**Multipartite Virus** A type of virus that is very infectious and can easily spread on your computer system. It can infect multiple parts of a system including memory, files, and boot sector which makes it difficult to contain.

Malware: Viruses, **WORMS**, Trojans, Logic Bomb, Bots, Rootkits

1. A **computer worm** is a standalone malware computer program that replicates itself in order to spread to other computers.
2. It often uses a computer network to spread itself, relying on security failures on the target computer to access it. It will use this machine as a host to scan and infect other computers
3. A computer worm has the ability to operate autonomously, without the need for a host file .
4. The computer worm does not usually infect computer files, but rather infects another computer on the network.

Malware: Viruses, **WORMS**, Trojans, Logic Bomb, Bots, Rootkits

How does a Computer Worm work?

In order to spread, computer worms use vulnerabilities in networks. The worm is looking for a back door to penetrate the network unnoticed. To get computer worms into circulation for the first time, hackers often send phishing e-mails or instant messages with malicious attachments. Cyber criminals try to camouflage the worm so that the recipient is willing to run the program. For this purpose, for example, double file extensions are used and / or a data name that looks harmless or urgent, such as “invoice”. When the user opens the attachment or link, they will immediately download the malware (computer worm) into the system or be directed to a dangerous website. In this way, the worm finds its way into the user’s system without them noticing. Once executed, the worm seeks a way to replicate and penetrate other systems. One way of doing this, for example, is for the worm to send an email to all contacts on the infected computer, which contains replicas of the worm.

Malware: Viruses, WORMS, Trojans, Logic Bomb, Bots, Rootkits

How does a Computer Worm work?

Many worms now have what is known as a payload. Payload in this case is an attachment that the worm brings with it. The worm can, for example, **CARRY RANSOMWARE** , viruses or other malware, which then cause damage to the infected systems. These can then, for example, delete files on the PC or encrypt files in the event of a blackmail attack. A computer worm can also install a back door that can later be exploited by other malware programs. This vulnerability gives the worm's author control over the infected computer.

Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

1. A Trojan horse, or Trojan, is a type of malicious code or software that looks legitimate but can take control of your computer.
2. A Trojan is designed to damage, disrupt, steal, or in general inflict some other harmful action on your data or network.
3. A Trojan acts like a genuine application or file to trick you. It seeks to deceive you into loading and executing the malware on your device. Once installed, a Trojan can perform the action it was designed for.
4. It can cause problems like killing background system processes, deleting hard drive data and corrupting file allocation systems.

Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

Trojan malware example

You might think you’ve received an email from someone you know and click on what looks like a legitimate attachment. But you’ve been fooled. The email is from a cybercriminal, and the file you clicked on — and downloaded and opened — has gone on to install malware on your device.

When you execute the program, the malware can spread to other files and damage your computer.

Malware: Viruses**,** Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

Types of Trojans

Backdoor Trojan: This Trojan can create a "backdoor" on your computer. It lets an attacker access your computer and control it. Your data can be downloaded by a third party and stolen. Or more malware can be uploaded to your device.

Distributed Denial of Service (DDoS) attack Trojan : This Trojan performs DDoS attacks. The idea is to take down a network by flooding it with traffic. That traffic comes from your infected computer and others.

Downloader Trojan: This Trojan targets your already-infected computer. It downloads and installs new versions of malicious programs. These can include Trojans and adware.

Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

Types of Trojans

Fake AV Trojan: This Trojan behaves like antivirus software, but demands money from you to detect and remove threats, whether they're real or fake.

Game-thief Trojan: The losers here may be online gamers. This Trojan seeks to steal their account information.

Infostealer Trojan: As it sounds, this Trojan is after data on your infected computer.

Mailfinder Trojan: This Trojan seeks to steal the email addresses you've accumulated on your device.

Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

Ransom Trojan
This Trojan seeks a ransom to undo damage it has done to your computer. This can include blocking your data or impairing your computer's performance.

Remote Access Trojan
This Trojan can give an attacker full control over your computer via a remote network connection. Its uses include stealing your information or spying on you.

Rootkit Trojan
A rootkit aims to hide or obscure an object on your infected computer. The idea? To extend the time a malicious program runs on your device.

SMS Trojan
This type of Trojan infects your mobile device and can send and intercept text messages. Texts to premium-rate numbers can drive up your phone costs.

Trojan IM
This Trojan targets instant messaging. It steals your logins and passwords on IM platforms.
That's just a sample. There are a lot more.

Malware: Viruses**,** Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

1. A logic bomb virus is a computer virus that contains a logic bomb, which is malicious code that triggers an attack when specific conditions are met.

2. ***Positive conditions*** refer to something happening, like a program opening, while ***negative conditions*** refer to something not happening, like someone not logging in.

3. Logic bombs are often installed by someone with high-level access, such as a system administrator. Such a person can cause mayhem by setting up logic bombs on multiple systems and programming them to "blow up" simultaneously when a certain event occurs, like when an employee is removed from the company's salary database.

Malware: Viruses, Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

1. A logic bomb virus is a computer virus that contains a logic bomb, which is malicious code that triggers an attack when specific conditions are met.

2. ***Positive conditions*** refer to something happening, like a program opening, while ***negative conditions*** refer to something not happening, like someone not logging in.

3. Logic bombs are often installed by someone with high-level access, such as a system administrator. Such a person can cause mayhem by setting up logic bombs on multiple systems and programming them to "blow up" simultaneously when a certain event occurs, like when an employee is removed from the company's salary database.

4. Another name for a logic bomb is **slag code,** which refers to the manipulated code that makes an otherwise safe program harmful.

Malware: Viruses**,** Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

All logic bombs share the following characteristics:

1. They lie dormant until triggered.
2. They carry an unknown payload, which is the part of the code that performs the attack.
3. They deliver the payload when a certain condition is met.

Among other things, a logic bomb can deliver its payload when:

1. A specified amount of time elapses.
2. A specific date occurs.
3. A certain transaction is processed.
4. A particular program opens.
5. Someone (for example, an admin) fails to log in.

A logic bomb's potential payload may be designed to:

- Corrupt data.
- Wipe hard drives.
- Delete files.
- Siphon off funds.
- Gather sensitive data.

Malware: Viruses, Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

A typical use case for a logic bomb is an insider attack. For example, let's say a privileged user has a grudge against his company and is afraid he might soon be fired. In advance of his firing, he sets up a scheduled job that checks to see if his user account has been active during the past 90 days; if no activity is found, the scheduled job deletes a critical database. Sure enough, the user is fired, and a few months later, once his account has been inactive for 90 days, the database is deleted. In most cases this would be a visible and obvious action, but what makes a logic bomb especially insidious is that it changes its code randomly, making it more difficult to detect and more damaging to the targeted organization.

Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

1. A rootkit is a malicious software that allows an unauthorized user to have privileged access to a computer and to restricted areas of its software.

2. A rootkit may contain a number of malicious tools such as keyloggers, banking credential stealers, password stealers, antivirus disablers, and bots for DDoS attacks. This software remain hidden in the computer and allow the attacker remote access to the computer.

Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

1. The term rootkit is derived from the combination of two words – "root" and "kit".
2. "Root" refers to the administrator account in Unix and Linux operating systems, which is an all-powerful account with full privileges and unrestricted access. It is equivalent to the administrator account in Windows systems.
3. The term "kit" refers to the programs that allow a threat actor to obtain unauthorized root/admin-level access to the computer and restricted areas.
4. The rootkit enables the threat actor to perform all these actions surreptitiously without the user's consent or knowledge.

Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

How the Attacker Installs Rootkits.

The attacker tries to obtain root/administrator access by exploiting known vulnerabilities, or by stealing administrator privilege credentials. Cyber criminals employ social engineering techniques to obtain credentials. Root access allows installation of rootkits or any other malware. Installation of the rootkit enables the threat actor to access the computer from remote to install other malware, steal data, observe activities and even control the computer. Rootkits are sophisticated malware, and most antivirus solutions and antimalware solutions do not detect rootkits. Rootkits are also able to hide their intrusion, and hence once they are in, they are practically undetectable.

Since rootkits have complete control over the system, they can modify software and the cyber security solutions such as the antivirus that could detect rootkits. As even the detection solutions are modified, it is difficult to detect and remove rootkits.

Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, ROOTKITS

What are Rootkits used for?

Rootkits are used for many puroses:

Stealth capabilities: Modern rootkits add stealth capabilities to malicious software payloads (such as keyloggers and viruses) to make them undetectable.

Backdoor access: Rootkits permit unauthorized access through backdoor malware. The rootkit subverts the login mechanism to also accept a secret login access for the attacker. Standard authentication and authorization mechanisms are bypassed to provide admin privileges to the attacker.

DDoS attacks: Rootkits allow the compromised computer to be used as a bot for distributed-denial-of-service attacks. The attack would now be traced to the compromised computer and not to the attacker's system. These bots are also called as zombie computers and are used as part of bot networks to launch the DDoS attacks, and other malicious activities such as click fraud and spam email distribution.

Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

The functionality of rootkits is also used for good causes, such as:

- in a honeypot to detect attacks
- to enhance emulation software
- to enhance security software – it enables the software to secure itself from malicious actions
- digital rights management enforcement
- device anti-theft protection - BIOS-based rootkit software enables monitoring, disabling and wiping of data on mobile devices when they get lost or stolen

There are five types of rootkits

1. **User-mode rootkits**
2. **kernel-mode rootkits**
3. **bootkits**
4. **hypervisor rootkits**
5. **firmware rootkits.**