**Linux Assignment**

1>Different types of hackers?

Sol: **White hat hackers:** Are those who are professionally expert in cybersecurity, they are authorized or certified to hack on the systems. They work for government or organizations by getting into the system. The hacking is done to test the level of cybersecurity in the organization, by this they can identify the weak points and fix them to avoid external attackers. White hat attackers work as per the rules and regulations set by government.

**Black hat hackers:** They are expert in cybersecurity but with wrong intentions. They attack other system to get access to system where they do not have authorized entry/ access. Hacker will steal the data or destroy the systems, this may make hacker a criminal.

**Gray hat hackers:** the intention behind the hacking is considered while categorizing the hackers. Gray hat hackers fall in between the black and white hat hackers. They are not certified. Their intention may be good or bad, if the intention is for personal gain, then the hacker is considered to be a gray hat hacker.

**Green hat hackers:** These are the hackers who are learning the ropes of hacking, the intention is to strives and learn to become full-fledged hackers. They look for opportunities to learn from experienced hackers.

**Blue Hat hackers:** the intention of this is to learn the missing, they use hacking as a weapon to gain popularity among their fellow beings. They use hacking to settle scores with their adversaries.

**Red hat hackers:** they are similar as white hackers, there intention is to stop the attack of black hat hackers. Sometimes red hat hackers are quite ruthless while dealing with black hat hackers or counteracting with malware. Red hat hackers continue to attack and may end up having to replace the entire system set up.

**Script kiddies:** They are amateurs' type of hacker in the field of hacking, they try to hack the system with scripts from other fellow hackers. They try to hack the system, network or websites. They do hacking for attention purpose and also doesn’t have complete knowledge about hacking.

2>What is different kind of attackers?

Sol: **Denial of services:** an attack on system’s resources, but it is launched from a large number of other host machines that are infected by malicious software controlled by the attacker.. Another purpose of a DoS attack can be to take a system offline so that a different kind of attack can be launched.

**Malware:** Malicious software can be described as unwanted software that is installed in your system without your consent. It can attach itself to legitimate code and propagate; it can lurk in useful applications or replicate itself across the Internet.

Types:

* **Macro viruses** — These viruses infect applications such as Microsoft Word or Excel. Macro viruses attach to an application’s initialization sequence.
* **File infectors** — File infector viruses usually attach themselves to executable code, such as .exe files. The virus is installed when the code is loaded.
* **System or boot-record infectors** — A boot-record virus attaches to the master boot record on hard disks. When the system is started, it will look at the boot sector and load the virus into memory, where it can propagate to other disks and computers.
* **Polymorphic viruses** — These viruses conceal themselves through varying cycles of encryption and decryption.
* **Stealth viruses** — Stealth viruses take over system functions to conceal themselves. They do this by compromising malware detection software so that the software will report an infected area as being uninfected.

**Phishing:** practice of sending emails that appear to be from trusted sources with the goal of gaining personal information or influencing users to do something. It combines social engineering and technical trickery. It could involve an attachment to an email that loads malware onto your computer.

**Drive by download:** Drive-by download attacks are a common method of spreading malware. Hackers look for insecure websites and plant a malicious script into HTTP or PHP code on one of the pages. This script might install malware directly onto the computer of someone who visits the site, or it might re-direct the victim to a site controlled by the hackers.

**Man-in-middle:** A MitM attack occurs when a hacker inserts itself between the communications of a client and a server. Here are some common types of man-in-the-middle attacks: session hijacking and IP spoofing.

**Password attack:** passwords are the most commonly used mechanism to authenticate users to an information system, obtaining passwords is a common and effective attack approach. Access to a person’s password can be obtained by looking around the person’s desk, ‘‘sniffing’’ the connection to the network to acquire unencrypted passwords, using social engineering, gaining access to a password database or outright guessing.

the last approach can be done in either a random or systematic manner:

* **Brute-force** password guessing means using a random approach by trying different passwords and hoping that one work Some logic can be applied by trying passwords related to the person’s name, job title, hobbies or similar items.
* In a **dictionary attack,** a dictionary of common passwords is used to attempt to gain access to a user’s computer and network. One approach is to copy an encrypted file that contains the passwords, apply the same encryption to a dictionary of commonly used passwords, and compare the results.

**Cross site scripting:** attacks use third-party web resources to run scripts in the victim’s web browser or scriptable application. Specifically, the attacker injects a payload with malicious JavaScript into a website’s database. When the victim requests a page from the website, the website transmits the page, with the attacker’s payload as part of the HTML body, to the victim’s browser, which executes the malicious script.

**Eavesdropping attacks:** they occur through the interception of network traffic. By eavesdropping, an attacker can obtain passwords, credit card numbers and other confidential information that a user might be sending over the network. Eavesdropping can be passive or active:

* **Passive eavesdropping** — A hacker detects the information by listening to the message transmission in the network.
* **Active eavesdropping** — A hacker actively grabs the information by disguising himself as friendly unit and by sending queries to transmitters. This is called probing, scanning or tampering.

**Birthday attacks:** they are made against hash algorithms that are used to verify the integrity of a message, software or digital signature. A message processed by a hash function produces a message digest (MD) of fixed length, independent of the length of the input message; this MD uniquely characterizes the message.

3> what are interfaces in java

Sol: An interface in Java is a blueprint of a class. It has static constants and abstract methods.

The interface in Java is *a mechanism to achieve abstraction. There* can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java. Interfaces specify what a class must do and not how. It is the blueprint of the class.

An Interface is about capabilities like a Player may be an interface and any class implementing Player must be able to (or must implement) move ().

If a class implements an interface and does not provide method bodies for all functions specified in the interface, then the class must be declared abstract.

**interface** <interface name> {

// declare constant fields

// declare methods that abstract

// by default.

}

4> what are use of Lambda in java?

Sol: A function which can be created without belonging to any class, Lambda expressions are similar to methods, but they do not need a name and they can be implemented right in the body of a method.

Enables functional programming: All new JVM based languages take advantage of the functional paradigm in their code.

Readable and concise code: People have started using lambda expressions and reported that it can help to remove a huge number of lines from their code.

Easy-to-Use APIs and Libraries: An API designed using lambda expressions can be easier to use and support other API.

5> what is #pragma?

Sol: The #pragma in C is adirective that is provided by the C standard in order to provide extra required details to the C compiler. These extra details can be anything that was somehow not passed within the program or the code logic. Is used to turn on or off some features. This type of directives is compiler-specific i.e., they vary from compiler to compiler.

#pragma startup and #pragma exit: These directives help us to specify the functions that are needed to run before program startup(before the control passes to main ()) and just before program exit (just before the control returns from main()).

#pragma warn Directive: This directive is used to hide the warning messages which are displayed during compilation. This may be useful for us when we have a large program and we want to solve all the errors before looking on warnings then by using it we can focus on errors by hiding all warnings. we can again let the warnings be visible by making slight changes in syntax.

Syntax:

#pragma warn +xxx (To show the warning)#pragma warn -xxx (To hide the warning)#pragma warn .xxx (To toggle between hide and show)

We can hide the warnings as shown below:

#pragma warn -rvl: This directive hides those warning which are raised when a function which is supposed to return a value does not return a value.

#pragma warn -par: This directive hides those warning which are raised when a function does not uses the parameters passed to it.

#pragma warn -rch: This directive hides those warning which are raised when a code is unreachable. For example: any code written after the return statement in a function is unreachable.

#pragma GCC poison: This directive is supported by the GCC compiler and is used to remove an identifier completely from the program. If we want to block an identifier then we can use the #pragma GCC poison directive.

#pragma GCC dependency: The #pragma GCC dependency allows you to check the relative dates of the current file and another file. If the other file is more recent than the current file, a warning is issued. This is useful if the current file is derived from the other file, and should be regenerated.

#pragma GCC dependency "parse.y"#pragma GCC dependency "/usr/include/time.h" rerun fixincludes

#pragma GCC system\_header: This pragma takes no arguments. It causes the rest of the code in the current file to be treated as if it came from a system header.

#pragma once: The #pragma once directive has a very simple concept. The header file containing this directive is included only once even if the programmer includes it multiple times during a compilation. This is not included in any ISO C++ standard. This directive works similar to the #include guard idiom. Use of #pragma once saves the program from multiple inclusion optimization.

Syntax:

#pragma once