**Day1 Linux Assignment**

1. Different Types of Hackers?

1). Red Hat hackers:

* Red hats are the types of hackers who often choose to take aggressive steps to stop black hat hackers.
* Infecting the bad hackers’ systems with malware.
* Launching DDoS attacks.
* Using tools to gain remote access to the hacker’s computer to demolish it.

2). White Hat Hackers:

* Companies and government agencies hire white hats as information security analysts, cybersecurity researchers, security specialists, penetration testers, etc.
* Find and fix vulnerabilities in the system before black hat hackers exploit them.
* Develop tools that can detect cyberattacks and mitigate or block them.
* Strengthen the overall security posture of the software and hardware components.
* Build security software like antivirus, anti-malware, anti-spyware, honeypots, firewalls, etc.

3). Grey Hat hackers:

* Grey hat hackers draw the owner’s attention to the existing vulnerabilities. They often launch the same type of cyber-attacks as white hats on a company/government servers and websites.
* These attacks expose the security loopholes but don’t cause any damage.
* Fix bugs or vulnerabilities,
* Strengthen the organization’s security defenses, or
* Provide recommendations, solutions, or tools to patch vulnerabilities.

4). Black Hat Hackers:

* Black hat hackers are the evil guys who want to use their technical skills to defraud and blackmail others.
* Sending [phishing emails](https://sectigostore.com/blog/what-is-a-phishing-email-5-examples-of-phishing-emails-and-how-to-avoid-them/) and [SMS messages](https://sectigostore.com/blog/what-is-sms-spoofing-how-can-you-prevent-it/).
* Writing, distributing, and selling [malware](https://sectigostore.com/blog/different-types-of-malware/) like viruses, worms, [trojan horses](https://sectigostore.com/blog/what-is-a-trojan-horse-in-computer-terms/), etc.
* Deploying [cyber attacks](https://sectigostore.com/blog/10-different-types-of-cyber-attacks-how-they-work/) like [distributed denial of service](https://sectigostore.com/blog/ddos-attack-statistics-a-look-at-the-most-recent-and-largest-ddos-attacks/) (DDoS) to slow down or crash the websites.
* Executing financial fraud and identity theft-related crimes.
* Deploying dangerous cyber threats like [brute-force attacks](https://aboutssl.org/what-is-brute-force-attack-how-to-prevent-it/), scareware, [botnets](https://sectigostore.com/blog/botnet-attacks-what-is-a-botnet-how-does-it-work/), [man-in-the-middle attacks](https://www.thesslstore.com/blog/protecting-against-man-in-the-middle-attacks/), [malvertizing](https://sectigostore.com/blog/what-is-malvertising-examples-and-protection-tips/) campaigns, etc.

1. What are the Different types of Attacks?

### DoS and DDoS Attacks, MITM Attacks (Man-in-the-middle), Phishing Attacks, Whale-phishing Attacks, Spear-phishing Attacks, Ransomware, Password Attack, SQL Injection Attack

### URL Interpretation, DNS Spoofing, Session Hijacking, Brute force attack, Web Attacks

### Insider Threats, Trojan Horses, Drive-by Attacks, XSS Attacks, Eavesdropping Attacks

### Birthday Attack, Malware Attack.

1. What are Interfaces in Java?

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*](https://www.javatpoint.com/abstract-class-in-java). There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple [inheritance in Java](https://www.javatpoint.com/inheritance-in-java).

Uses of Interfaces in Java: -

* It is used to achieve abstraction.
* By interface, we can support the functionality of multiple inheritance.
* It can be used to achieve loose coupling.

Syntax of Interface Declaration:

interface <interface\_name>

{

// declare constant fields

// declare methods that abstract

// by default.

}

1. What is the use of Lambda Expression in Java?

Lambda expressions basically express instances of [functional interfaces](https://www.geeksforgeeks.org/functional-interfaces-java/) (An interface with single abstract method is called functional interface. An example is java.lang.Runnable).

lambda expressions implement the only abstract function and therefore implement functional interfaces.

lambda expression helps in achieving the internal iteration of collections rather than external iteration.

Syntax:

lambda operator -> body

1. What is the Switch Case Replacement for Python?

Python does not have a switch or case statement. To get around this fact, we use [dictionary mapping](https://www.youtube.com/watch?v=z7z_e5-l2yE).

Example Function to convert number into string Switcher is dictionary data type here

def numbers\_to\_strings(argument):

switcher = {

0: "zero",

1: "one",

2: "two",

}

return switcher.get(argument, "nothing")

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

argument=0

print (numbers\_to\_strings(argument))

1. What is Preprocessor directive in C?

The preprocessor directives give instruction to the compiler to preprocess the information before actual compilation starts. All preprocessor directives begin with #, and only white-space characters may appear before a preprocessor directive on a line.

Some Examples of Preprocessor:

#define: It defines a sequence of characters, called symbol.

#undef: It allows you to undefine a symbol.

#if: It allows testing a symbol or symbols to see if they evaluate to true.

#else: It allows to create a compound conditional directive, along with #if.

#elif: It allows creating a compound conditional directive.

1. What is #pragma in C?

It is used to provided additional information to the Compiler. The #pragma in C is a directive 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.

Syntax:

#pragma token\_name

Some of the Pragma Directives:

#pragma startup: Before the execution of main(), the function specified in pragma is needed to run.

#pragma exit: Before the end of program, the function specified in pragma is needed to run.

#pragma warn: Used to hide the warning messages.

#pragma GCC dependency: Checks the dates of current and other file. If other file is recent, it shows a warning message.

#pragma GCC system\_header: It treats the code of current file as if it came from system header.

#pragma GCC poison: Used to block an identifier from the program.