FAST School of Computing

NATIONAL UNIVERSITY OF COMPUTERS AND EMERGING SCIENCES

ISLAMABAD CAMPUS

Assignment # 1

Kisaa Batool	**20i-1829**
Aleena Fatima Khalid	**20k-1688**

SE-6Q

Information Security CS3002

Question # 1:

- **Principle of least privilege**

The Java Code that follows principle of Least Privilege is as follows:

```
Public deleteFromCart(int uid,itemID){
If(customer.isCustomer(custId) == true && item.isInCart==true){
        Item.remove();
System.out.println("Deleted item from cart!");
}
else if(customer.isCustomer(custId) == false && item.isInCart==false){
System.out.println("You can't delete such item");
exit();
}
}
```

The Java Code that does not follow principle of Least Privilege is as follows:

```
Public deleteFromCart(int uid,itemId){
If(customer.isCustomer(custid))
     Item.remove();
System.out.println("Deleted Item from cart!");
}
```

In the first coding example the item that can be deleted from the cart is the one which is in the cart and user can delete the item from his cart only while in the other code the user can enter the customer id of any customer and delete items from his cart here is violation of principle and user is given access which is not required for him.

- **Principle of fail-safe defaults**

The Java Code that follows principle of fail-safe defaults is as follows:

```
Public TransferFunds(double funds, receipientId){
If(userLoggedInCorrectly() == true){
If(currentBalance>funds && recipientId==correct){
TransferFunds(funds);
System.out.println("Transfer Successfully!");
}
else{
}
}
System.out.println("Insufficient Funds!");
else if(LogInStatus() == false){
System.out.println("Account disabled for transferring funds");
exit();
}
}
```

The Java Code that does not follow principle of fail-safe defaults is as follows:

```
Public TransferFunds(double funds){
TransferFunds();
System.out.println("Transferred money from account!");
}
```

The first code has an if statement to check that it can work only if account is logged in correctly and allow to transfer funds if there are enough current amount as well in account.

The second code has no fail-safe defaults checks and allows all accounts to transfer
cash without checking their status which violates the principle of fail-safe defaults.

- **Principle of economy of mechanism**

The Java Code that follows principle of economy of mechanism is as follows:

```
Public simpleLibrary(){
Public void issueBook(String book id){ book id;
};
Public void returnBook(String bookId, String readerId)
{
BookId status update;
Reader status update;
};
Public void searchBook( String name){Traverese all book database;
Return book relavant details;};
}
```

The Java Code that does not follow principle of economy of mechanism is as follows:

```
Public simpleLibrary(){
Public void issueBook(String book id){ book id;
};
Public void returnBook(String bookId, String readerId)
{
BookId status update;
Reader status update;
};
Public void searchBook( String name){Traverese all book database;
Return bookdetails;};
}
// additional functionalities that add complexity
Public void showBooksAccordingToUserSearches(){}
Public void addToWishList(){}
```

Public void readerLastSearches(){}
The first code includes the minimum number of functionalities that is more than enough
for one of the simplest libraries. There is less complexity involved and few chances of
errors.
The second code violates the principle because it adds additional functionalities that are unnecessary for the simple bank and causes makes the
app more complex and prone to risks and errors.

- **Principle of complete mediation**

- **Principle of separation of privileges**

- **Principle of least common mechanism**

The Java Code that follows principle of least common mechanism is as follows:

```
Public loginBankApp(){
Public void BiometricAuthenticate(String recognition)
{if(recog==true) loginStatus=true;}
If(loginStatus ==true)
{ System.out.println("Successfully Logged in!");}
Else
{ System.out.println("Login Denied!");}

updateBalance(String name, double balance)
{update.balance;
System.out.println("Update Done!");}
```

The Java Code that does not follow principle of least common mechanism is as follows:

```
Public login(){
Public loginBankApp(){
Public void BiometricAuthenticate(String recognition)
{if(recog==true) loginStatus=true;}
If(loginStatus ==true)
{ System.out.println("Successfully Logged in!");

updateBalance(String name, double balance)
{
update.balance;
System.out.println("Update Done!");}}
Else
{ System.out.println("Login Denied!");}
```

The first code follows the principle of least common mechanism since the biometric authentication and update balance functionalities are not shared. The login status does not automatically interfere with the updating process.
The second code violates the principle because it gives a shared automatic access to updating the balance. A vulnerability in login status can allow the attacker to bypass it and
modify the bank balance of the user.

Question # 2:

No.	Mechanisms	Principles
1.	Hardware Security Module	• **Principle of Separation of Privilege**: HSM ensures that only authorized users are granted access, for eg using a cryptography key transfer and their isolation from other functions. • **Principle of Fail-Safe Default:** Since by default, typical HSM denies access if explicit authorization is not provided, especially at the time of crash or failure. • **Principle of Economy of Mechanism:** HSM does not support gold requirements, or extra functionalities. They specialize in their key management operations, as well as cryptography transfers. They don't complicate the process, and keep it simplified with their focus on certain security objectives and functions.
2.	Cuckoo Sandbox For Malware Analysis	• **Principle of Open Design:** Cuckoo sandbox for malware analysis is an open-source project, meaning it has its design open for speculation, and vulnerabilities in the system are expertly managed and processed. • **Principle of Fail-Safe Default:** Cuckoo sandbox works in an isolated, and secure environment, as well as with the use of virtual machines to adhere to security. With their default mechanisms and configuration, they achieve maximum security. • **Principle of Separation of Privilege:** Since it runs in an isolated environment, e.g. VMs, it separates the malware with the system.
3.	Access Control List In An OS	• **Principle of Least Privilege:** With the OS's guidelines, access controls lists only provide resources to the approved users or groups. They can limit the access as write only/read only etc, according to the rights required by that particular user. • **Principle of Complete Mediation:** Since OS's access control lists adhere to specific rules, they check every time when a particular user requires access for a certain resource, and if that access is not allowed to them, it is denied.

		• **Principle of Fail-Safe Default:** Since with OS' access control list, you have to explicitly allow access to resources, the default configuration is to deny access until authorized.
4.	Image Captcha On Flex	• **Principle of Economy of Mechanism:** The image captcha on flex is an extremely simplified process, and does not risk any complexities. • **Principle of Psychological Acceptability:** The image captcha on flex is extremely user-friendly and easy to solve. It does not compromise security, but it also manages to provide ease to the user, and is accepted widely by the users of Flex.
5.	Password Strength Indicator	• **Principle of Psychological Acceptability:** Password indicators, whether on google, or other websites are user-friendly and are widely accepted by users due to the ease they provide, so they fall under this principle.
6.	Biometric Authentication Required Before Banking App	• **Principle of Complete Mediation:** When biometric verification is performed every time the user requests access for the banking app, it is explicitly checked that the user is authorized. • **Principle of Least Privilege:** With the use of extensive biometric authentication, it is clear that access is only given to the rightful person, and nothing extra. It is only given the right to authorize the user's identity, and nothing extra that may compromise security on user's data and such.
7.	AES Encryption Ciphers	• **Principle of Open Design:** Due to its extremely sensitive nature, particular details are not shared with the public, but that standard design and implementation is open to community for speculation. • **Principle of Economy of Mechanism**: According to the given design and implementation of the AES encryption ciphers, they are relatively simple and comprise no complexities, falling under this principle.
8.	Atomicity In Database Transactions	• **Principle of Least Privilege:** This principle can be applied to this security mechanism, considering the access and authorization rights it allows before in the transaction process.
9.	Intrusion Detection Systems In Front Of Public Facing Servers In Organizations	• **Principle of Fail-Safe Default:** Due to its default nature of taking the network traffic as malicious unless specified otherwise, IDS systems fall under this category due to their effective security measures.

Question # 3:

In the CVE the main security principle that is failed to uphold is the principle of the least privilege. As the principle states that that a system or user should only have access to the resources and information necessary to perform their specific tasks, and no more. However, in this weakness gave an aggressor to acquire raised rights in a weak climate, which thusly conceded them admittance to additional assets and data than they ought to have had. This might actually prompt further double-dealing of the framework or association.

The guideline of Least privilege is urgent in light of the fact that it restricts the potential harm that can be brought about by hackers and unauthorized access to confidential data. At the point when this rule is dismissed, weaknesses like CVE-2007-0408 can make critical security gambles for the individuals as well as organizations.

Another most likely violation is of principle of complete mediation since it is related with adding extra steps for authorization to ensure hackers don't make an unauthorized access and to validate real user. But this CVE fails to validate certificates of the clients when reusing cached connections. It at last permits hackers to utilize untrusted X.509 certificate to get unapproved admittance to the framework. This is all a direct result of absence of intervention in legitimate checking and confirmation of client's character. There should be a middle person framework to intercede the appropriate checking and confirmation of the client's character to forestall hackers for unapproved access.

Question # 4:

- **Air-gapping of important machines/servers in companies:**

Principle of least common mechanism refers to isolation of users and their activities from each other. This states that users should not share processes, threads and information channels between uses. While air gapping computer is physically segregated and incapable of connecting wirelessly or physically with other computers or network devices. Air gaps protect critical computer systems or data from potential attacks ranging from malware and ransomware to key loggers or other attacks from malicious actors.

This is why we can say that this enforces this security principle since significant data is stored on these computers and servers, and they need to be expertly secured against illegal access and malicious manipulation. Businesses can lower the risk of intrusion or assault by totally isolating them from the internet and other networks by air-gapping them. For devices and servers that manage crucial infrastructure, like those used in the banking, energy, and defense industries, this strategy is extremely crucial.

Air-gapping does have certain drawbacks, though. Remotely accessing and managing these computers and servers may become more challenging as a result and can be a pressure to deal with. However, air-gapping is not totally considered safe according to security studies, because attackers can still employ physical or social engineering tactics to reach these separated workstations. As a result, it's crucial to apply additional security measures in addition to air-gapping.

- **Cloudflare protection for websites:**

The guideline of least common mechanism expresses that components used to get to assets should not be shared. Sharing assets gives a channel along which data can be communicated, thus such sharing must be limited to ensure proper security. On the off chance that the working framework offers help for VMS, the working framework will uphold this honor in its default manner. If not, it will offer some help, i.e. a virtual memory space, but not complete help.

For example, a site gives electronic business administrations to a significant organization. Attackers need to deny the organization of the income they get from that site. They flood the site with messages, and tie up the electronic business administrations. Genuine clients can't get to the site and, thus, take their business somewhere else. Here, the sharing of the Web with the attackers' locales made the attack succeed. The proper countermeasure is to limit the aggressors to achieve the business goal? Methods to do this incorporate intermediary servers, for example, the Purdue SYN intermediary3 or traffic choking. The previous targets suspect associations; the last option lessens load on the pertinent portion of the organization unpredictably.

- **The Colonial Pipeline ransomware attack:**

The Colonial Pipeline ransomware assault is a disturbing illustration of how the security design of least common mechanism was dismissed, prompting crushing weaknesses for the association. The attackers had the option to take advantage of a weakness in a solitary framework that approached the whole organization, permitting them to convey the ransomware all through the association and encode information on numerous frameworks. This defect in the framework configuration abused the principle of least common mechanism, which goes against the utilization of similar component for numerous reasons and suggests that any common components should be straightforward and secure.

The utilization of a solitary framework with admittance to the whole organization empowered the attackers to sidestep a significant number of the association's safety efforts, highlighting the significance of planning secure frameworks that limit the effect of a solitary compromised part. By sticking to the principle of least common mechanism, associations can lessen the dangers of broad attacks and better protect themselves from progressively modern digital dangers. In outline, the Colonial Pipeline ransomware attack is a distinct sign of the basic significance of executing strong safety efforts, and the critical outcomes that can arise because of dismissing sound security standards.