# **Procedure CDP09- IT Security**

# **Risk Assessment report**

## **SQL Injection:**

SQL injection is a kind of code injection that the attacker can destroy or get access to the database. This is one of the most common website hacking techniques. SQL injection is malicious code injection through input fields in the website.

SQL injection can be done in different ways, like:

**Based on 1=1:** The attacker gives input in the input field in such a way that the condition gets true, it can be cleared more through an example:

*Select \* from users where userId = 1 or 1=1*

Let suppose if someone wants to get unauthorized access to the login page of the system, for which data is required from the users table, the above query can be used as SQL injection as 1=1 is always true, the table provides data of the users.

## **Protect SQL Injection:**

The easiest way to protect your database from SQL injection is to parametrize your SQL query. Do not use SQL command directly on the inputs, take the data, keep it in other variables and use that for the SQL query.

## **Password Encryption:**

While taking password from the user, it can be saved in encrypted form in the database. The system takes the input in such a way that the admin of the database himself is not able to check the password, the password is stored in the database in code words. At the time of insertion in database, it is encrypted and at the time of accessing the password, it is decrypted again. We will be using password encryption.

## **Validate Input fields:**

Malicious codes and other inputs can also be harmful for the database and the software systems. JavaScript functions and other checks can be used to prevent such inputs and validate them. Checks like limiting the input sizes and defining inputs. We will be using these checks to keep our system safe.

## **Session Management:**

Sometimes we login to our system from any other person’s phone or PC and forget to logout, lose access to internet or electricity due to which our account remains logged in that system. Sessions can be used to protect the system from such kind of risks like if the system is not used for a specific interval of time, it must log out automatically when the page Is refreshed. Or if the browser is closed the account must log out.

## **Cross site Scripting:**

Cross-site scripting (also known as XSS) is a web security vulnerability that allows an attacker to compromise the interactions that users have with a vulnerable application. It allows an attacker to circumvent the same origin policy, which is designed to segregate different websites from each other. Cross-site scripting vulnerabilities normally allow an attacker to masquerade as a victim user, to carry out any actions that the user is able to perform, and to access any of the user's data. If the victim user has privileged access within the application, then the attacker might be able to gain full control over all of the application's functionality and data.

## **Server Misconfiguration:**

Server misconfigurations means that the attackers try to bypass the authentication. Many servers come with unnecessary default and sample files, including applications, configuration files, scripts, and webpages. They may also have unnecessary services enabled, such as content management and remote administration functionality. Debugging functions may be enabled or administrative functions may be accessible to anonymous users. Servers may include well-known default accounts and passwords. Failure to fully lock down or harden the server can leave improperly set file and directory permissions.

## **Denial of Service attacks:**

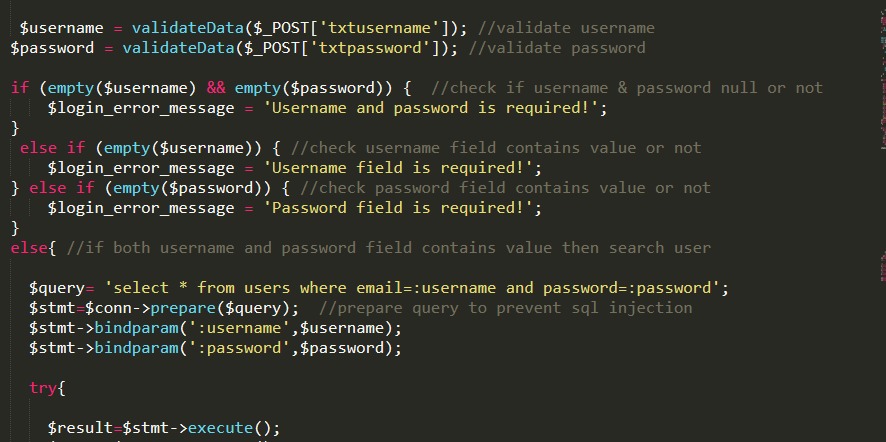
In such kind of attacks, the attacker actually takes your system out of service for the real users. They bombard thousands of requests for data on your system so the system get out of action. The server crashes because of such malicious requests.

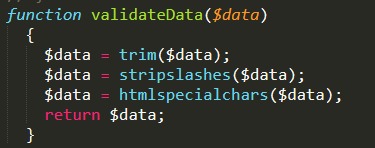
# **Implementation:**

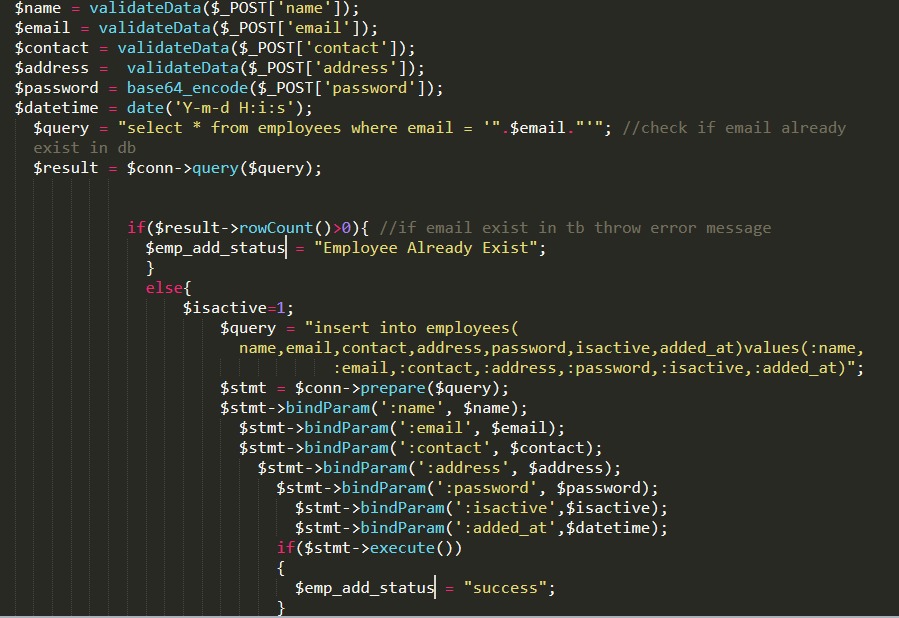
Implementation of the said security risks is done to keep the system safe.

## **SQL Injection and input validation:**

Following code shows the implementation of SQL injection prevention and validate input:

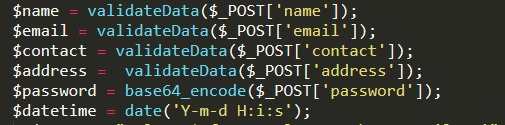






## **Password Encryption:**

The base64\_encode is a function that encrypt the password



## **Session management:**

