ASSIGNMENT – 1 25/03/2023

# Understanding OWASP Vulnerabilities

## A01:2021-Broken Access Control

Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.

**Vulnerability name:** Exposure of Information through Directory Listing

**CWE:** CWE-548

**Description:** A directory listing provides an attacker with the complete index of all the resources located inside of the directory. The specific risks and consequences vary depending on which files are listed and accessible.

**Business Impact:** This can give the attacker access to customer and clientele information. This can inevitably lead to data breaches and loss of customer privacy. This may look like a greater problem for the user, but the company is the one that pays the fine for the loss. The company also faces damage to reputation.

**Vulnerability name:** Cross-Site Request Forgery

**CWE:** CWE-352

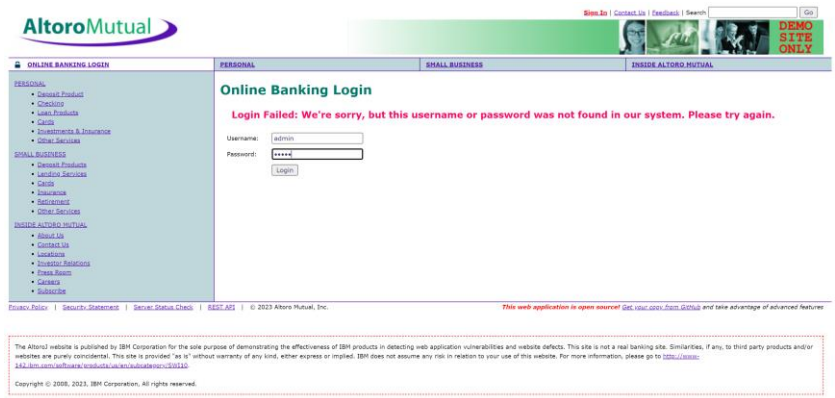
**Description:** When a web server is designed to receive a request from a client without any mechanism for verifying that it was intentionally sent, then it might be possible for an attacker to trick a client into making an unintentional request to the web server which will be treated as an authentic request. This can be done via a URL, image load, XML Http Request, etc. and can result in exposure of data or unintended code execution.

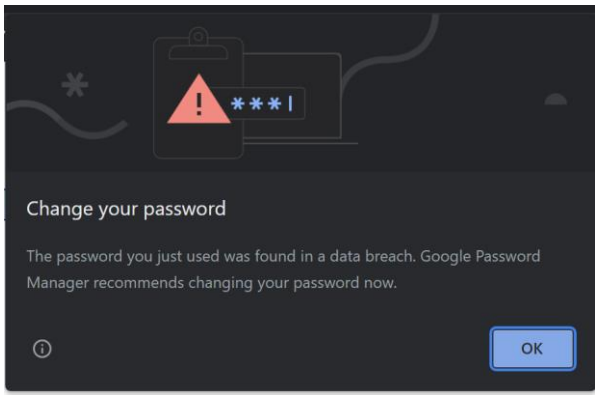
**Business Impact:** If the victim of this attack ends up being an important person like the administrator of the web application or anyone with substantial amount of credentials and authority, the attacker ends up gaining full or parts of control which is inexcusable. This leads to a lot of problems where there can be disruptions in service and financial loss based on the attacker’s intentions.

## A02:2021-Cryptographic Failures

The first thing is to determine the protection needs of data in transit and at rest. For example, passwords, credit card numbers, health records, personal information, and

business secrets require extra protection, mainly if that data falls under privacy laws, e.g., EU's General Data Protection Regulation (GDPR), or regulations, e.g., financial data protection such as PCI Data Security Standard (PCI DSS).


**Vulnerability name:** Use of Cryptographically Weak Pseudo-Random Number

**CWE:** CWE-338

**Description:** When a non-cryptographic PRNG is used in a cryptographic context, it can expose the cryptography to certain types of attacks.

Often a pseudo-random number generator (PRNG) is not designed for cryptography. Sometimes a mediocre source of randomness is sufficient or preferable for algorithms that use random numbers. Weak generators generally take less processing power and/or do not

use the precious, finite, entropy sources on a system. While such PRNGs might have very useful features, these same features could be used to break the cryptography.

**Business Impact:** Access Control can be easily lost. ID or cryptographic key can be easily accessed. They can gain access to functionalities which was previously restricted. This leads to loss of control over the web application which can be maliciously used to victimize innocent users.

**Vulnerability name:** Use of RSA Algorithm without OAEP

**CWE:** CWE-780

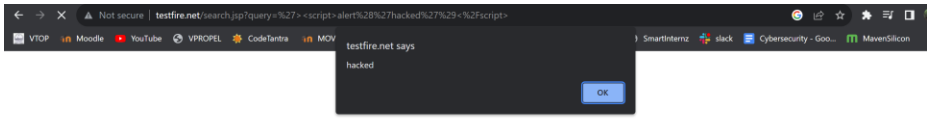
**Description**: Padding schemes are often used with cryptographic algorithms to make the plaintext less predictable and complicate attack efforts. The OAEP scheme is often used with RSA to nullify the impact of predictable common text.

**Business Impact:** If your organization's security practices are questioned due to encryption weaknesses, it could give competitors an advantage in terms of customer trust and market credibility. Remediation efforts to address encryption vulnerabilities could disrupt regular business operations and divert resources away from strategic initiatives.

## A03:2021-Injection

An application is vulnerable to attack when:

- ✓ User-supplied data is not validated, filtered, or sanitized by the application.
- ✓ Dynamic queries or non-parameterized calls without context-aware escaping are used directly in the interpreter.
- ✓ Hostile data is used within object-relational mapping (ORM) search parameters to extract additional, sensitive records.
- ✓ Hostile data is directly used or concatenated. The SQL or command contains the structure and malicious data in dynamic queries, commands, or stored procedures.




**Vulnerability name:** XML Injection

**CWE:** CWE-91

**Description:** Within XML, special elements could include reserved words or characters such as "<", ">", """, and "&", which could then be used to add new data or modify XML syntax.

**Business Impact:** Attackers might modify XML data to introduce malicious content, which could lead to altered business logic, incorrect processing of data, or unauthorized changes to system behaviour.

## A04:2021-Insecure Design

Insecure design is a broad category representing different weaknesses, expressed as “missing or ineffective control design.” Insecure design is not the source for all other Top 10 risk categories. There is a difference between insecure design and insecure implementation. We differentiate between design flaws and implementation defects for a reason, they have different root causes and remediation. A secure design can still have implementation defects leading to vulnerabilities that may be exploited. An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks. One of the factors that contribute to insecure design is the lack of business risk profiling inherent in the software or system being developed, and thus the failure to determine what level of security design is required.

**Vulnerability name:** Improper Use of Validation Framework

**CWE:** CWE-1173

**Description:** Many modern coding languages provide developers with input validation frameworks to make the task of input validation easier and less error-prone. These frameworks will automatically check all input against specified criteria and direct execution to error handlers when invalid input is received. The improper use (i.e., an incorrect

implementation or missing altogether) of these frameworks is not directly exploitable, but can lead to an exploitable condition if proper input validation is not performed later in the product. Not using provided input validation frameworks can also hurt the maintainability of code as future developers may not recognize the downstream input validation being used in the place of the validation framework.

**Business Impact:** It involves the misuse of validation frameworks or libraries, potentially leading to vulnerabilities in software applications. These vulnerabilities could result in data breaches, unauthorized access, or system disruptions, leading to compromised customer trust, reputational damage, legal and regulatory consequences, financial losses due to incident response and recovery efforts, and operational disruption due to remediation, all of which can collectively harm the organization's competitiveness and long-term viability.

## A05:2021-Security Misconfiguration

The application might be vulnerable if the application is:

- Missing appropriate security hardening across any part of the application stack or improperly configured permissions on cloud services.
- Unnecessary features are enabled or installed (e.g., unnecessary ports, services, pages, accounts, or privileges).
- Default accounts and their passwords are still enabled and unchanged.
- Error handling reveals stack traces or other overly informative error messages to users.
- For upgraded systems, the latest security features are disabled or not configured securely.
- The security settings in the application servers, application frameworks (e.g., Struts, Spring, ASP.NET), libraries, databases, etc., are not set to secure values.
- The server does not send security headers or directives, or they are not set to secure values.

**Vulnerability name:** ASP.NET Misconfiguration: Creating Debug Binary

**CWE:** CWE-11

**Description:** ASP .NET applications can be configured to produce debug binaries. These binaries give detailed debugging messages and should not be used in production environments. Debug binaries are meant to be used in a development or testing environment and can pose a security risk if they are deployed to production.

**Business Impact**: Poses a notable business impact by leaving software, systems, or applications vulnerable due to improperly configured security settings. This can lead to

unauthorized data access, breaches, service disruptions, financial losses from incident response and recovery, damage to reputation and customer trust, regulatory violations, and diverting resources from strategic initiatives to rectify security gaps, thereby undermining operational efficiency and competitiveness.

**Vulnerability name:** Cleartext Storage of Sensitive Information in a Cookie

**CWE:** CWE-315

**Description:** Attackers can use widely-available tools to view the cookie and read the sensitive information. Even if the information is encoded in a way that is not human-readable, certain techniques could determine which encoding is being used, then decode the information.

**Business Impact:** Allows sensitive data to be stored in an unprotected, easily accessible format. This could lead to data breaches, unauthorized access, identity theft, legal and regulatory penalties, reputational damage, financial losses due to incident response, customer distrust, and operational disruption caused by remediating the issue. The resulting negative outcomes can harm the organization's credibility, customer relationships, and overall business viability.