**Sensitive Data Exposure**

This attack could happen when an unauthorized person gets access to a sensitive information and abuses it, when accidentally clicking a suspicious link, downloading malicious malware etc. Information security and data privacy may in result be violated with this attack. Applications can be vulnerable when: data is stored in plain text, there is a lack of HTTPS on authenticated pages, hashed passwords that can still be easily cracked, and when tokens are disclosed in public source code.

**How to Prevent**

To prevent such attacks, security measures should be effectively implemented. It is crucial to encrypt data and define accessibility. Data must be encrypted in both stored or transit form because any data in the form of open text serves as an invitation for attackers. This can be implemented by enforcing key based encryption- identifying which data requires extra protection and limit access to only legitimate users. To prevent such attacks, use secure authentication. It also helps to make sure that you use a strong password by applying a password hashing function algorithm, changing your password from time to time, and not using the same password for different platforms. Conducting regular risk assessment and updating the security system can also counter potential threats. Lastly, it is important to keep a protected and secure backup of sensitive data so that when an attack is successful, it is still possible to mitigate the losses.

**Injection**

An injection attack happens when an attacker relay almost any source of data that can be used to manipulate malicious codes into the user’s system. The source of data can be a parameter, an internal web service as well as external web service, an environment variable, an injection vector and lastly, all types of users. It is easy to discover the vulnerabilities of the code with the help of scanners and fuzzers. Common languages that are common to such vulnerabilities are Structured Query Language, Lightweight Directory Access Protocol, NoSQL queries, Operating System commands, Extensible Markup Language parsers, Simple Mail Transfer Protocol headers, expression languages and Object-relational Mapping queries. Common effects of such attacks are loss of data, stolen privacy information, corruption, accountability loss, denial of access to data.

The attacker may know if the user’s application is vulnerable to attack when the user supplied some data that aren’t sanitized by the application. Directly usage or concatenation of data, extraction of sensitive data by object-relational mapping search parameters.

**How to prevent**

It is preferred to use an API that is safe which avoids the provision of a parameterized interface or usage in interpreter entirely. For applications that does not require special characters, using a positive server-side input validation can prevent such attacks. When also querying data, add LIMIT option and SQL control queries to avoid other data to be hacked by the attacker.

**Broken Authentication**

This happens when an attacker does something to the authentication or session management in order to steal data from the accounts or use the accounts and act as the owner. Attackers could easily compromise passwords, keys, or session tokens because most application function related to authentication and session management are not correctly implemented. Most flaws are in logout, password management, timeout etc.

**How to Prevent**

People should be careful in making their passwords. Their minimum password length should be 8 characters long. Also their passwords should be consist of alphanumeric characters and conventional symbols and is not easy to guess. When there is an authentication error such as invalid password or invalid username, the message that should appear is “Invalid username and/or password” instead of “Invalid username” or “invalid password”. Account should be disabled temporarily in case of number of invalid login attempts for protection against brute force login. However, it shouldn’t be long enough also for denial-of-service attack.