**SQL Injection**

**Blind SQL Injection:**

Blind SQL Injection is a type of SQL Injection attack where the attacker is unable to directly see the results of the injected SQL queries but can infer information about the database indirectly. This is typically achieved by observing the behavior of the application's responses to crafted SQL queries. Blind SQL Injection attacks can be classified into two main types: Boolean-Based Blind SQL Injection, where the attacker infers information based on whether the application's response is true or false, and Time-Based Blind SQL Injection, where the attacker induces delays in the application's responses to extract information.

**Time-Based SQL Injection:**

Time-Based SQL Injection is a type of SQL Injection attack where the attacker uses time delays to infer information about the database indirectly. This is accomplished by injecting SQL queries that contain time-delaying functions or commands (e.g., `SLEEP()` in MySQL, `WAITFOR DELAY` in SQL Server) into vulnerable input fields or parameters. By analyzing the application's response times to these injected queries, the attacker can infer information about the database structure, data, or perform reconnaissance to identify potential injection points.

**Boolean Exploited SQL Injection:**

Boolean Exploited SQL Injection is a type of SQL Injection attack where the attacker exploits vulnerabilities in a web application's database layer by crafting SQL queries with boolean logic operators (e.g., `AND`, `OR`, `NOT`). By constructing conditional statements that control the application's behavior based on injected conditions, the attacker can infer sensitive information about the database structure or contents. This type of attack aims to manipulate the application's responses to reveal hidden information.

**Heavy Query SQL Injection:**

Heavy Query SQL Injection is a type of SQL Injection attack where the attacker executes complex and resource-intensive SQL queries to overwhelm the database server. By injecting queries that are computationally expensive or consume significant resources (e.g., nested queries, complex joins, aggregate functions), the attacker aims to degrade the performance of the database server, potentially leading to denial of service (DoS) conditions or performance degradation for legitimate users.

**In-Bind SQL Injection:**

It refers to a type of SQL Injection attack where the attacker uses the same communication channel to both launch the attack and gather results. In-Band SQL Injection attacks occur when the attacker is able to directly retrieve the results of the injected queries over the same communication channel used to perform the injection. This type of attack includes techniques such as error-based SQL Injection and Union-based SQL Injection, where the attacker receives immediate feedback from the application regarding the success or failure of the injected queries.

**Error-Based SQL Injection:**

Error-Based SQL Injection is a type of SQL Injection attack where the attacker exploits error messages generated by the database server to infer information about the database structure or contents. By injecting malicious SQL queries that intentionally trigger errors (e.g., syntax errors, type mismatches), the attacker can observe detailed error messages returned by the application, which may contain sensitive information such as database schema, table names, or query results.

**Union-Based SQL Injection:**

Union-Based SQL Injection is a type of SQL Injection attack where the attacker leverages the `UNION` SQL operator to combine the results of multiple queries into a single result set. By injecting a crafted SQL query containing a `UNION` operator into vulnerable input fields or parameters, the attacker can append arbitrary SQL code to an existing query, allowing them to retrieve additional information from the database. Union-Based SQL Injection attacks are commonly used to extract data from database tables that are not directly accessible through the application's interface.

**End-of-Line Comment SQL Injection:**

End-of-Line (EOL) Comment SQL Injection is a type of SQL Injection attack that exploits the presence of comment characters (`--` or `#`) to bypass input validation mechanisms and execute malicious SQL queries. In SQL, comments are used to annotate SQL code and are ignored by the database server during query execution. Attackers can leverage end-of-line comments to append arbitrary SQL code to legitimate queries, thereby altering their behavior and potentially compromising the security of the application.

**Piggybacked Query Injection:**

Piggybacked Query Injection is a sophisticated form of SQL Injection attack where an attacker exploits vulnerabilities in a web application's database layer to execute multiple SQL queries in a single injection attempt. This technique allows the attacker to perform a sequence of malicious operations within the context of a single vulnerable query, potentially bypassing security controls and gaining unauthorized access to sensitive data or performing unauthorized actions.

**System Stored SQL Injection:**

System Stored SQL Injection is a type of SQL Injection attack that occurs when an attacker exploits vulnerabilities in a web application's database layer to execute malicious SQL code stored within the database system itself. Unlike traditional SQL Injection attacks that involve injecting code directly into application input fields, System Stored SQL Injection leverages stored procedures, functions, or other database objects containing malicious SQL code to manipulate the application's behavior and compromise data integrity.

**Out-of-Band SQL Injection:**

Out-of-Band (OOB) SQL Injection is a type of SQL Injection attack where the attacker exploits vulnerabilities in a web application's database layer to exfiltrate data or perform actions using an alternate communication channel rather than directly receiving the results within the application's response. Unlike traditional SQL Injection attacks that rely on the application's response to extract information, OOB SQL Injection leverages out-of-band channels such as DNS requests, HTTP requests, or SMTP requests to transmit data or execute commands.