Database Administration Security

Cybersecurity Department
Course Code: ***CBS 214***
Theoretical Lecture 5: SQL Injection

For further understanding and deeper insight into the concepts discussed in this document, **students are strongly encouraged to watch the accompanying video lectures provided along with the lecturer's notes available on Microsoft Teams**. These resources will help reinforce the theoretical knowledge with practical explanations and real-world examples.

Injection attacks are among the most serious security risks affecting web applications today. According to the OWASP Top 10 reports, injection vulnerabilities have consistently ranked among the most critical threats, holding the first position in 2017, dropping to third place in 2021, and ranking fifth in 2026. Despite this shift in ranking, injection attacks particularly SQL injection remain highly dangerous due to their ease of exploitation and potentially severe consequences.

SQL injection is a type of attack that targets databases by inserting malicious SQL code into input fields such as login forms, search boxes, or URL parameters. This attack becomes possible when a web application fails to properly validate or sanitize user input before including it in an SQL query. Instead of treating the input as data, the application unintentionally allows it to be executed as part of the SQL command. As a result, attackers can manipulate database queries to gain unauthorized access, retrieve sensitive information, or even modify and delete data.

In a typical SQL injection scenario, an attacker carefully crafts input that alters the structure of a database query. When the web application processes this input, it constructs an SQL statement and sends it to the database. Since the database trusts the application, it executes the query without recognizing that it contains malicious instructions. The database then returns the results back through the application to the attacker, often exposing information that was never intended to be publicly accessible.

SQL injection continues to be a major concern for several reasons. First, it is highly discoverable, meaning vulnerabilities can often be found with minimal effort, sometimes by simply entering special characters into input fields. Second, it is widespread, as many web applications especially older ones still contain insecure coding practices. Third, it is easy to exploit, with numerous automated tools available that can perform attacks with little technical expertise. Finally, the impact of a successful SQL injection attack can be severe, potentially leading to full database compromise, data breaches, financial losses, and damage to an organization's reputation.

The mechanics of a SQL injection attack follow a standard process. An attacker sends a specially crafted HTTP request to a web application. The application processes this request and builds a database query using the provided input. This query is then sent through the system often passing through firewalls without issue because it appears to be legitimate traffic and executed by the database. The database returns the results, which may include sensitive or unauthorized data, back to the attacker via the web application.

An important aspect of SQL injection lies in how web applications authenticate with databases. Typically, the application uses a single set of stored credentials to connect to the database, regardless of the user making the request. If these credentials have excessive privileges, an attacker exploiting SQL injection can perform a wide range of harmful actions, including reading confidential data, modifying records, or executing administrative commands. This misuse of privileges significantly increases the severity of the attack.

SQL injection vulnerabilities can exist across various technologies, including programming languages such as ASP.NET, PHP, and Java, as well as database systems like SQL Server, MySQL, and Oracle. While modern development frameworks have introduced features that help prevent such vulnerabilities, older systems remain particularly at risk. Applications built with outdated technologies often lack proper security controls, making them frequent targets for attackers.

In real-world scenarios, SQL injection attacks have led to significant data breaches and system compromises. Organizations affected by such attacks may suffer not only financial losses but also legal consequences and a loss of customer trust. For this reason, understanding SQL injection is essential for both developers and cybersecurity professionals.

In conclusion, SQL injection is a critical and ongoing threat in web security. It arises primarily from poor input validation and insecure coding practices. Due to its simplicity, prevalence, and potentially devastating impact, it is essential to implement proper security measures, including input validation, parameterized queries, and least-privilege database access. By understanding how SQL injection works, individuals can better protect systems and reduce the risk of exploitation.