**Essay Paper: SQL Injection Attack**

**Introduction:**

SQL injection attack is one of the most common attack vectors. The attack depends on malicious SQL code for accessing data, which was not previously intended for display. The attacks occur with the help of backend database manipulation, and the information displayed may include sensitive company information, including the lists of private customers and their personal data. Businesses and private corporations have fallen prey to the attacks, which sometimes cause databases and loss of sensitive information. Attackers may also target company databases, gain administrative rights, and steal customer information, including credit card details, phone numbers, and residential addresses. The attackers insert arbitrary SQL into their chosen companies' databases to either delete, modify, or copy database content.

**How SQL Injection Attacks Occur:**

SQL attacks start with identifying the database and injecting SQL queries through the client's input data to the application. Therefore, the client must first type their information into the database for the attack to commence. Successful attacks lead to the exploitation of sensitive information through data modification (Thiyab & Basil, 2017). The attackers then tamper with the recorded information and sometimes cause repudiation problems, including changing balances and voiding transactions. Depending on the attacks' motives, the attackers may choose to delete database information or take over as the new administrators of the database. The most common attacks occur through ASP and PHP applications since cases. The attacker's skills and experience determine the level of damage they can cause to the database and the effectiveness of the SQL attacks.

The attacks begin on the page where database users add their information. Once the users create their pages and insert query runs to include the user information in the database, the users can log into the application using a password and a username. A select query then runs for the retrieval of the user information. The attacker can then insert information that alters or changes query and force the query to return unexpected results to the user. For example, if the customer's username in a database sis Ken12, the attacker may type in "OR 1=1-"at the username. The quotation that begins the attacker's query ends the username parameter and makes a new query search for the user as a blank space or blank quotes (Lawal, Sultan & Shakiru, 2016). The attacker then types OR1=1 after the quotation, and the hyphen commands the query to comment out and ignore the other texts. When the query gets the user's information, it becomes easy for the attacker to recall data, especially when the customer or database user does not insert a username into the database.

**Injection Through User Input:**

This injection mechanism occurs when an attacker introduces malicious SQL statements into a vulnerable application. The attackers inject commands crafted user input after confirming that the user input is suitable for the targeted web application. Web applications have different ways of reading user information, depending on the type of environment or conditions under which they are deployed. In most SQL applications that use cookies, user information comes from submissions from the users sent to the internet application via HTTP request or GET posts. The attacker knows that the web applications can access all the variables in their environment, and that is how they target the web applications for SQL attacks.

**Injections through Cookies:**

Cookies are state files generated by internet applications and recorded on the customer's computers and other devices. When the user logs in or returns to a particular internet application, cookies store their state information unless the client decides to delete the cookies on their devices. SQL attackers take advantage of the client's control over the cookies to tamper with client information. Web applications that use cookies data to build SQL queries are the most vulnerable since all the attacker has to do is submit an SQL attack by embedding the attack in the cookies (Basit, Hendawi, Chen & Sun, 2019). The types of attacks are commonly used to attack companies with web applications for customers to create user accounts and record their personal information.

**Server-variable injections:**

Server variables are various variables that have network headers, HTTP, and environmental variables. There are multiple ways web applications use the variables, including identifying user trends and logging usage statistics. If the user logs in to a particular database without sanitization, SQL injection attacks could be targeted at the user's information or account (Loughran, Salih & Subburaj, 2018). SQL attackers can manipulate the values in the network headers and HTTP, enabling them to place SQL attacks directly into the headers. When the system issues a query to log to the server variable, the request triggers the manipulated header's attack.

**SQL Injection in Bank Transaction:**

Banking institutions and customers are some of the most vulnerable entities to SQL injections. Attackers may target a WIFI network in the streets or the restaurant and commence the attack. They run different utilities and redirect a person's information through their devices. The attacker then sniffs the data and acts as a certificate server for SSL. The certificate acts as a means of establishing a secure SSL connection. If the user had a reliable certificate and connects directly to a particular website, the certificate will decrypt your data from the browser to the SSL website. The bank's website will then use the information from the certificate to decode the customer information. The issue becomes a problem for any hackers who may be planning to steal customer credentials.

Suppose the user has a fake certificate, which is usually sent by the hacker and connects directly through the device instead of connecting directly to the baker's website. In that case, this becomes a critical vulnerability. The user's data gets transmitted between the browser and the attacker's device, and the attacker successfully grabs the data traffic. The attacker is the issuer of the certificate in this case, and therefore, they can decrypt customer data using the same certificate.

**SQL Injection Attacks on online retailers:**

Malicious hackers take advantage of the massive data traffic between online sellers and customers. They target poorly coded internet applications that are mostly used by new online retailers and their customers. The SQL attack begins when the attacker introduces malicious codes into an online retailer's network and systems. New websites are vulnerable since most of them do not invest in reliable web applications and security. Therefore, internet applications do not adequately validate or filter clients' information when opening a new account and with credentials. The attacks occur when a customer orders for a product through a retailing web application such as Alibaba. They send malicious SQL queries to the database to access information and plant various codes on the network. Most of the SQL injection attacks on online retailers and businesses occur through the user-input mode.

**Dangers of Injection Attacks:**

SQL injection attacks can be devastating for private and public corporations. Internal and external sensitive information can be sued to commit severe cybercrimes, including financial fraud and identity theft. The attacks affect the companies from which the data gets stolen and the owners of the information. The company's reputation declines, and customers avoid such companies' fear of losing sensitive information to the same attackers. Social security number is one of the essential information that an SQL attack can disclose to an attacker. The SSN can be sued to commit a crime over a long period without the owner's knowledge since the number does not get old. Bank information and credit card information need timely action since the owner or user may note changes in their transactions, block the cards, or report to the authorities.

SQL injection attacks are detrimental to companies and their customers.

Attackers take advantage of various vulnerabilities, including poorly designed web applications and security vulnerabilities on websites and company systems. Attackers use different mechanisms, including user input, cookies, and server-variable methods. The data and client credentials stolen from SQL attacks can be used in different ways, including identity theft, where the hackers use customer information for criminal activities. Banks and online retail businesses are some of the most vulnerable institutions due to many daily transactions. Companies that operate through Internet applications should invest in security to safeguard company systems and customer credentials.

**Conclusion:**

In conclusion, there are different ways at which SQL injections attacks can occur. For example, it may occur through cookies, user input or even through server variable. These attacks can have devastating impacts on organizations, banking institutions and customers. In fact, some sensitive data or information may be lost and could end up being used for identity theft. Therefore, there is need for them to invest in reliable security to protect themselves against these attacks.

**References:**

Basit, N., Hendawi, A., Chen, J., & Sun, A. (2019, February). A learning platform for SQL injection. In *Proceedings of the 50th ACM technical symposium on computer science education* (pp. 184-190).

Lawal, M. A., Sultan, A. B. M., & Shakiru, A. O. (2016). Systematic literature review on SQL injection attack. *International Journal of Soft Computing*, *11*(1), 26-35.

Loughran, D. T., Salih, M. K., & Subburaj, V. H. (2018, August). All About SQL Injection Attacks. In *Journal of The Colloquium for Information System Security Education* (Vol. 6, No. 1, pp. 24-24).

Thiyab, R. M., Ali, M. A., & Basil, F. (2017, April). The impact of SQL injection attacks on the security of databases. In *Proceedings of the 6th International Conference of Computing & Informatics* (pp. 323-331).