Joshua Powell

CSS332

Unit 4

Individual Project

SQLi (structured query language injection)

SQL is the backbone language of any database. This includes a lot of companies that exist today. They use databases to store their metadata along with a lot of other things that pertain to their company. When it comes to SQLi there are several ways an attacker can maliciously act on a database that can cause problems within the database that could in turn disrupt business operations. One such way is through what is called an in-band SQLi. In this type of injection attack, the attacker will use the UNION operator in SQL to try and get a Hypertext Transfer Protocol (HTTP) response from the database. This can be done by using the UNION operator in a combination of two or more select statements into one single statement to get that HTTP response from the database. Another injection technique is called inferential (blind) SQLi. This is where the attacker will use ways to query the database to see if they get a response to in turn study the database's behavior. The issue here is the attacker is trying to reconstruct the database via payloads and watching the Web Application's response to that payload. This is without seeing any of the in-band processes such as the actual network traffic and just watching the web application itself. There are several issues that can arise from SQL attacks when it comes to the business and the flow of operations. One can say with the right injection technique you could destroy a database. One damaging aspect of an injection attack is the leak of confidential data such as employees' names and addresses. There is also the chance that an attacker can get complete control over a database's data and change things around that should not be changed. There is also the chance that through a successful SQLi attack the attacker can

completely deface the web application giving a bad rep for the company and could possibly result in the company failing (Muscat, 2017).

Cross-site Scripting (XSS)

This type of attack is malicious like SQL injection; however, it is a little bit different. With Cross-site Scripting, the attacker inserts malicious code into the web application itself. What that means is they have basically reverse-engineered the software for the web application in a way that they are able to insert code into it and send a link to the user. They usually make it enticing by presenting it as a real link for something pertaining to the application. There are three main types of Cross-site Scripting. They are Reflected XSS, Stored XSS, and DOM-Based XSS.

1. Reflected XSS (AKA Non-Persistent or Type I) is when a user inputs a certain thing into the search browser of a web application, and it immediately returns some type of error message or searches, resulting in part or all the intended results by the user. This is without storing the user's data permanently. In some cases, the users provided data may never even leave the browser.

2. Stored XSS (AKA Persistent or Type II): With this type of cross-scripting the users' data is stored on the target server, such as a database, in a message forum, visitor log, comment field, etc. And then the victim can retrieve that data with the web application without rendering it safe in the browser (Types of XSS | OWASP Foundation, n.d.). With the implementation of HTML 5, we can envision that the attack payload will be stored within the victims' browser and never even be stored in the server at all.

3. DOM-Based XSS (AKA Type-0): This is where the tainted flow of data will proceed from the source i.e., the web application, to the sink and takes place within the browser. For example, the source where the malicious data is read could in fact be the URL of the page (e.g., document.location.href), or it could be in the element of the HTML, and the sink is a sensitive method called that causes the execution of the malicious code (e.g., document.write) (Types of XSS | OWASP Foundation, n.d.).

For years most thought of these as separate groups, but as of 2012 the development of two different categories stated that these three previous groups were thought to be of their category. They were found to overlap so they came up with server and client XSS which goes into more detail as to how each of the other groups overlaps each other (Types of XSS | OWASP Foundation, n.d.).

Individual Project 5:

How the attacks Occurred

Well, first, vulnerabilities are exploits just waiting to happen. It takes ingenuity and a lot of time for security teams to critique their profession and keep on updating their tactics depending on the changing landscape. There are a lot of ways that an attacker can get information to be able to do such attacks. They could also do their research and find all types of data just by simple google searches. This could in turn lead them to an SQL injection attack or a Cross-cite scripting attack that leads to the company falling into the hands of the attacker. Another issue to consider is the fact that these could in turn lead to other breaches if the data segments and the way they are utilized are the same. I for one find that the issue here is not necessarily the lack there of a good security posture but from research that companies who lack a

good patch program for their devices and software are the first to feel the brunt of a cyber-attack (Evidence Backing That Most Cyber Attacks Use the Simplest of Attacks - Zoeken, n.d.).

Problems that cause these attacks

We cannot bring up what went wrong without bringing up what caused what when wrong. When referring to any type of breach, exploit, or attack there was always a risk assessment before whether it was taken seriously or not shown in the attacks that were performed. Just following the simplest of risk adjustments or just implementing the simplest of controls could in turn prevent the attack. There are a lot of companies in shambles as we are halfway through 2022. It was reported this past June that in 2021 cybercrime cost American companies more than a whopping 6.9 billion dollars in costs. It comes down to companies and businesses taking the necessary precautions and adopting the appropriate policies to combat these risks we face today (Evidence Backing That Most Cyber Attacks Use the Simplest of Attacks - Zoeken, n.d.).

Damages of SQLi

SQLi (Sequential language injection) is not to be taken lightly as there are some serious repercussions if a successful SQLi is committed. One very serious repercussion is confidential data that a customer and they are leaked to the attacker. This can lead to lawsuits against the company that could in turn cause that company to fail as their policy did not adjust for such attacks and financial loss presented by this successful SQLi. Another thing to keep in mind when it comes to SQLi is that it can also compromise trade secrets or individual computers within the company. This can lead to a serious compromise for a company that depends on its trade secrets

to be profitable and its individual computers to keep the company up and running (Damages Done by SQL Injection - Zoeken, n.d.).

Damage of Cross-site Scripting (XSS)

This type of attack can bring a company to its' knees given the right implementation. A company depends on its customers and its products so it can remain functional and operational. If an attacker were to implement a successful XSS attack it could mean very bad news for the company. For example, let's just say that an attacker was able to gain access to companies' main site and was able to run an XSS attack on that web application. They could display false information and mislead the customers making it very hard for the customer support team because it is not true, yet the attacker was able to make adjustments that they were not supposed to be able to. One serious aspect of a successful XSS attack is the attacker having the ability to gain access to the companies' user accounts. In doing so they can get the users' passwords, bank account numbers, Usernames, etc., and use that information for financial gain, political motivation, and all other kinds of malicious activity. The only hold on what these types pose a threat to is the expanse of imagination. Any type of attack whether it was simple or not is a serious issue that should not be taken lightly by any company or organization (Cross Site Scripting (XSS) | OWASP Foundation, n.d.).

What should have been done?

A Company/Business needs to sit down and go over its overall risk assessment and risk management. A business these days is only as strong as its overall risk analysis and what they choose to be more president than the other risks present in the analysis. This being said, a company does not have to put in controls just because the risk exists, but the bigger question they

have to ask is does that risk outweigh the repercussions and cost to implement controls for that risk. Another thing that has to be taken into account when considering a risk analysis and the overall processes of the business, is whether they have a policy in place that goes over the risks involved with the work they do and what the employees need to do to guarantee the success of the company. With SQLi a company should implement a policy that goes over the risks associated with SQLi, second do not trust any user input look at it as always untrusted. Third use a whitelist and not a blacklist, fourth, Adopt the latest Technologies, and Fifth employed verified mechanisms (What Is SQL Injection (SQLi) and How to Prevent Attacks, 2022). When it comes to Cross-Site Scripting implementing the correct syntax is vital in ensuring the web application stays secure and sound. One such method is using "createElement()" and assigning property values with appropriate methods to prevent attackers from putting malware on sites and compromising the companies' business (Prevent Cross-Site Scripting (XSS) in ASP.NET Core, 2022). In the world of businesses today a company needs to be very careful but not so careful as to hinder the company, but they cannot just pretend that risks do not exist in their company, nor can they not take the necessary precautions in educating their employees on the risks of links they do not know or that having access to things they should not have access to is very very bad and should be reported. If they do not report it they are implicating themselves with the risks of having that access.

References

1. Muscat, I. (2017, October 12). *SQLi part 5: Inferential SQLi (Blind SQLi)*. Acunetix. Retrieved October 17, 2022, from https://www.acunetix.com/blog/articles/sqli-part-5-inferential-sqli-blind-sqli/
2. *Types of XSS | OWASP Foundation*. (n.d.). Retrieved October 19, 2022, from https://owasp.org/www-community/Types_of_Cross-Site_Scripting
3. *evidence backing that most cyber attacks use the simplest of attacks - Zoeken*. (n.d.). Retrieved October 24, 2022, from https://www.bing.com/search?q=evidence+backing+that+most+cyber+attacks+use+the+simplest+of+attacks
4. *damages done by SQL injection - Zoeken*. (n.d.). Retrieved October 24, 2022, from https://www.bing.com/search?q=damages+done+by+SQL+injection
5. *Cross Site Scripting (XSS) | OWASP Foundation*. (n.d.). Retrieved October 24, 2022, from https://owasp.org/www-community/attacks/xss/
6. *What is SQL Injection (SQLi) and How to Prevent Attacks*. (2022, July 21). Acunetix. https://www.acunetix.com/websitesecurity/sql-injection/
7. *Prevent Cross-Site Scripting (XSS) in ASP.NET Core*. (2022, June 3). Microsoft Learn. https://learn.microsoft.com/en-us/aspnet/core/security/cross-site-scripting?view=aspnetcore-6.0