CSCI 5409 - SECURITY & BUSINESS CONSIDERATIONS CRITICAL ANALYSIS AND RESPONSE

Group Name – Apes Together Strong

Instructor – Dr. Lu Yang

Dhrumil Amish Shah (B00857606)
Jaspreet Kaur Gill (B00879409)
Nachiket Niranjanbhai Panchal (B00882397)
Samiksha Narendra Salgaonkar (B00865423)

1. How does your application architecture keep data secure at all the layers? If it does not, if there are vulnerabilities, please explain where your data is vulnerable and how you could address these vulnerabilities with further work?
   Response: Our application architecture is divided into two layers – Frontend and Backend.
   Our team is trying to keep the data of the application secure by following some tactics such as:
   a) We are storing sensitive data such as usernames, passwords to the database by encrypting the data.
   b) Users cannot access any URL of the application directly. But, the URL of the application will redirect the user to the login page asking to enter the valid credentials to authenticate the user.
   c) We are not exposing sensitive information such as username, password, or any file link in the URL of the application while re-directing from one page to the other page of the application.

   The vulnerabilities that are present in these two layers are described in detail.
   Vulnerabilities at frontend layer –
   a) Cross-site request forgery (CSRF) – Cross-site request forgery is based on social engineering by which an attacker can make the authenticated user click on the link, image, or any URL to take control of the user's session and perform changes to the web application or steal the application data. So, it seems like the attack is carried out by the legitimate user and malicious requests are sent to the web application to exploit the website [1].
   b) CSS injection – This attack uses developers to use the CSS file containing malicious import statements. While building an application, numerous packages are required to install to clear the dependencies and at this point mistakes may occur. The developer may download the dependencies from the untrusted website and execute malicious code exploiting the website such as CSS keylogger leaks passwords to the attacker [1].
   c) XSS – Attackers can add the malicious code to the input fields present in the application and the application does not recognize the malicious and displays the exact input entered, which may lead to exposing the sensitive data of the application [1].
   Vulnerabilities at backend layer –
   a) SQL injection – SQL injection allows users to alter the backend database of the application. An attacker can execute the unintended commands to corrupt the database data or can read the user data such as username and password [2].
   b) Broken Authentication and Session Management – The website maintains session cookies that contain sensitive data of the user such as username and password. If cookies are not declared invalid when the session ends on logging out by the user or closing the browser abruptly, then the sensitive data would be present inside the session cookies. If the user is using any public computer, then the cookies will sit on that computer exposing the sensitive data to the attacker [2].

2. Which security mechanisms are used to achieve the data security described in question 1? List them, and explain any choices you made for each mechanism (technology you used, algorithm, cloud provider service, etc.)

   Response: For the vulnerabilities mentioned above can be prevented in our project as descried below solutions.

   Solutions at frontend layer –

   a) Cross-site request forgery (CSRF) – As the Cross-site request forgery happened due to session or cookie theft, we can simply add a random CSRF Token. The CSRF Token can be assigned for each session with a random string and verified with each request after login so no cross-site request can have authorization.
   b) CSS injection – The CSS injections happens most of the time when we use the un-trusted CSS links without any verification. The best solution we can apply is use the verified and self-hosted CSS libraries.
   c) XSS – The Cross-site scripting known as XSS attack is one of the most popular attacks. The XSS attacks can be prevented by sanitizing input values from front-end. Also, using un-trusted JavaScript libraries concludes into vulnerabilities, so using only trusted JavaScript libraries should be preference.

   Solutions at backend layer –

   a) SQL injection – The SQL injection is one of the most popular back-end vulnerability attack. Generally, while getting data from GET or POST method in back-end contains some parts of SQL which modifies the back-end query and results in data leaks. To prevent the SQL injections, we should sanitize strings for SQL keywords and any query breaking symbols like double-quotes ("), single-quotes ('), equals (=), and many more. Some languages like PHP provides inbuilt methods to sanitize strings for SQL injections.
   b) Broken Authentication and Session Management – This vulnerability happens due to unmanaged sessions and cookies after leaving a web application. The best solution to prevent this kind of vulnerability is to destroy sessions and cookies as soon as the user leaves the web application.

3. What would your organization have to purchase to reproduce your architecture in a private cloud while providing relatively the same level of availability as your cloud implementation? Try to give a rough estimate of what it would cost, don't worry if you are far off. These systems are complicated, and you don't know all the exact equipment and software you would need to purchase. Just explore and try your best to figure out the combination of software and hardware you would need to buy to reproduce your app on-premise.

   Response: Owning a private cloud and hosting the application on it is expensive. There are many different costs involved in owning a private cloud. The below list contains various services to be purchased and their associated costs:

   - Compute
   - Storage
   - Network
   - Security

   The associated costs would be as given in the table below

Services to purchase	Resource option available	Cost per month (CAD)
Server	CPU, RAM	Over $1000
Storage	HDD and SSD	Over $300
Network	Bandwidth, IPs	Over $80
Security	Data protection, Network protection, Application protection	Over $400

   Additionally, the architecture setup requires a one-time investment in setting up and configuring the project. Lastly, there will be maintenance cost incurred for supporting the overall setup.

4. Which cloud mechanism would be most important for you to add monitoring to in order to make sure costs do not escalate out of budget unexpectedly? In other words, which of your cloud mechanisms has the most potential to cost the most money?
   Response: The cloud services for our application travel memories that might possibly invoke higher costs are Amazon EC2, Amazon S3, and Amazon DynamoDB. These services consume high computation power and storage space. Thus, these services require monitoring. The other services in the application like Amazon ECS, Amazon ECR, and Amazon Cognito will only be charge as and when used at a very low pricing.

   Amazon EC2 is the most expensive cloud service of all the services utilized in Travel Memories application. Amazon EC2 runs throughout the day and even if the EC2 instance is stopped, it charges the storage cost for hosting the application on Elastic Block Store (EBS) used by Amazon EC2.

References

[1] D. E. Team, “Frontend security vulnerabilities: Why you should care,” *Debricked*, 18-Oct-2021. [Online]. Available: https://debricked.com/blog/frontend-security-vulnerabilities/. [Accessed: 20-Nov-2021].

[2] L. Williams, “10 most common web security vulnerabilities,” *Guru99*, 07-Oct-2021. [Online]. Available: https://www.guru99.com/web-security-vulnerabilities.html. [Accessed: 20-Nov-2021].