Available online www.jsaer.com

Journal of Scientific and Engineering Research, 2023, 10(5):379-385

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Security Measures for Front-End Applications: User Authentication and Authorization

Chakradhar Avinash Devarapalli

Software Developer
Email: avinashd7[at]gmail.com

Abstract The user experience is directly linked with the interface of the application and the exchange of information and actions happens with front-end interactions. To retain the user's trust, the unique identity of an individual needs to be identified for a dedicated session where only secured interaction is possible. User authentication is therefore important for authorized access into the system. The research document demonstrates the possible security measures to protect front-end applications from unauthorized access. The various security threats can be, SQL injections, vulnerable APIs, clickjacking, session hijacking, and other authorization and authentication threats. There are certain problems from the development end in creating a secure system. The solution suggested against these problems if implemented appropriately can mitigate the security breaches. This as a result can help the organization retain integrity and maintain a healthy relationship with users. The implications of future developments are even more reliable for achieving the goal of secured front-end applications.

Keywords user authentication, authorization, security, front-end applications, privacy, security challenges

1. Introduction

The front-end applications are mainly focused on the User Interfaces and are responsible for providing a smooth experience to users on web and mobile app platforms. The user experience is directly connected to developing more robust front-end applications. As the complexity of these applications increases with more features, it becomes difficult to maintain security. Authorizing user profiles with authenticated access to the system can help secure the system with appropriate measures. The unauthorized access can be avoided by adopting different authentication techniques. The security of the system for instance can be increased with the use of modern technologies like Angular which provides centralized authentication to the system [1].

The research article focuses on general security threats to front-end applications. Specifically, authentications and authorization threats are covered by their technical reasons. The possible solutions are suggested against the identified security threats to avoid malicious attacks. The industrial problems for security considerations are also discussed with their appropriate ways to avoid these issues. As Figure 1 shows, the weakest point of security where most violations take place is the exchange of information between the client side and server side. The recommended method to overcome this problem is therefore the use of JSON Web Tokens which provide secure data exchange with the help of JSON objects.

The JSON objects work on the authentication with the help of unique tokens attached to the requests. This helps in security improvement where the user data is at the most risk. The user experience can be enhanced with the effective use of the JWT technique [2].

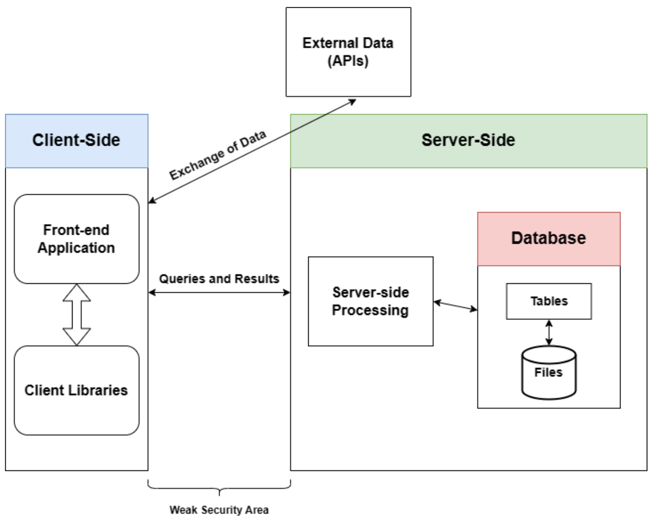

Figure 1: Architecture Diagram of Front-end Applications

Literature Review

A combination of authentication and authorization returns a strong system in terms of security. Robust access controls are provided for organizations with these two terminologies [3]. Only the authenticated individual can perform the allowed action within the boundary defined by the developers. The authorization comes next when permissions need to be granted to access the features by users within their role in the system. It minimizes the potential attacks and restricts the users from unauthorized access [4], [5].

The security measures for web applications are guided by the Open Web Application Security Project (OWASP). It is an organization that helps the large organization secure their system by considering the important factors that play a major role in protecting the system from unwanted activities [6].

The front-end applications can be secured with the help of encryption methods like HTTP. The HTTP encryption assists in secure communication between the front-end and back-end terminals and keeps the system updated with the latest implementations [7]. The logic attacks can be identified with different testing techniques like black-box testing [8].

Problem Statement

The front-end is the essence of an application and is responsible for gaining the interest of the user towards the product which is why millions of developers are employed around the world to make the front-end applications secure for both users and organizations. However, there are difficulties associated with securing front-end applications with the increased hierarchy of components involved within the system. This writing focuses on appropriate methods to avoid unauthorized access to front-end applications to avoid security and privacy breaches from both end users and service providers. It gives guidelines for the industry to overcome the challenges faced during the implementation of secured components in front-end applications.

Security Threats

The Front-end applications can be targeted with different means and the most famous of these attacks are covered here,

Cross-Site Request Forgery

Users are forced to take harmful actions making it look legitimate. It mainly occurs on web applications where users have logged in to the system [9].

Cross-Site Scripting

The user data is attacked with this technique when attackers target the user identity by injecting scripts. These malicious injections in webpages identify the web pages visited by other users [9].

SQL Injections

The behavior of the form fields is studied in these attacks, loopholes are identified and then they modify the input field properties to achieve the intrusion. The SQL injections are the most famous of these and the databases are targeted with forced violations. However, it can only happen if there are security holes in the database attached with form fields. For instance, consider the following query,

*SELECT * FROM users WHERE username = '' OR '1'='1';*

Vulnerable APIs

The use of deprecated external components with the use of old APIs can help achieve the vulnerability here. This is because any previous version of the software has loopholes that can be targeted if it's not continuously updated.

Clickjacking

The appearance of pages is mainly targeted under these attacks. They make the external forms or sections look the same as a part of the system being used but actually, it belongs to a scammed setup.

Storage Vulnerabilities

The storage space can be targeted with the help of files accompanied by attacking scripts. It is also possible through the use of cookies attached to the browser of the client-side system.

Authentication Threats

The commonly noticed authentication issues are,

Session Attacks

The user sessions can be targeted if they are not properly secured with appropriate tokens.

Two-Factor Authentication

The two-factor authentication helps to add an extra layer of security and if not used can be targeted [10].

Password Breakage

The weak or easily guessed passwords become ignorance from the user end. The password saved somewhere in the storage without any encryption or in a browser that can be attacked is also responsible for the system getting attacked.

Broken Access Controls

If an application is developed with no boundaries and the user can explore the unnecessary files then there are high chances of attacks.

Authorization Threats

The authorization threats can appear if the privileges of authenticated users are targeted. The possible examples can be,

Unnecessary Privileges

If there are more privileges provided than needed, other accounts don't remain secured as these could be otherwise.

Lack of User Privileges

The common permissions if not granted can lead to bad user experiences.

Injection Attacks

These can be used to target form fields where a security gap is available for the attackers.

Improper Access

The access if given inappropriately, an unauthorized user can violate the policies.

Inappropriate distribution

The resources if distributed randomly can invite a user to target other users on the system using the access provided.

Solutions

The possible solutions to these security threats as well as authentication and authorization threats are [11],

- Input Validations to prevent attacks like SQL Injections.
- Implementing OAuth2.0 protocol for front-end authorization.
- Use Content Security Policy and Anti-CSRF tokens.
- Restrict malicious cookies on client-side browsers.
- Provide access based on diverse user roles.
- Carry Regular Updates and Security Audits.
- Implement security headers to prevent attacks like session hijacking and clickjacking.
- Use appropriate protocols like HTTPS.
- Encrypt sensitive data and secure client-side storage.
- Avoid weak passwords and use two-factor authentication.

Apart from these generalized solutions, there is a term called JSON Web Tokens which allows to transfer of data in a secured environment. The JSON Object is used to transmit information from one end and receive from the other and it allows secure transferring with digitally signed data.

JSON Web Tokens

The data between two parties like server and client can be exchanged securely with the help of .JSON Web Tokens where data is present in the form of objects in a JSON file. These objects are not only secure but easily accessible as well. The JSON web tokens are encoded and are divided into header, payload, and signature [2]. The header gives information about the algorithm type and token type, the payload refers to the actual data, and the third component signature refers to the secret key which preserves the integrity of information.

The Flow of JSON Web Tokens (JWT) is given in Figure 2 below which ensures secure token exchange and user authentication based on the unique token,

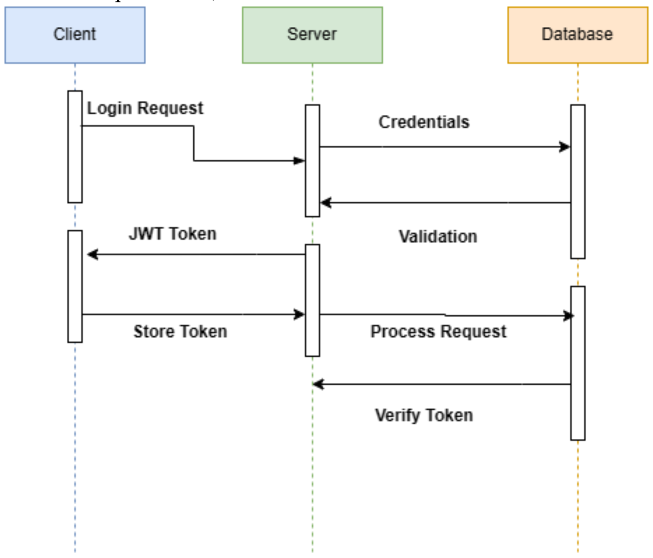

Figure 1: JSON Web Tokens Sequence Flow

This can be implemented with the following algorithm where a secret key is required to be added by the developer,

```
const jwt = require('.JSON Web Token');
const key = 'Secret Key Here';

// Create JWT
const payload = { KEY: USER_ID(INT) };
const token = jwt.sign(payload, key, { expiresIn: 'TIME' });
// Verify token
jwt.verify(token, key, (error, decoded) => {
  if (!error) {
    console.log('Decoded token:', decoded);
  }
    return;
});
```

Industrial Challenges

The developers tend to face some challenges while maintaining the security of front-end applications when working on the implementation. The most common problems are,

Maintaining User Experience

The balance between better user experience and implementing complex security components is difficult to maintain. It's because the system needs to be user-friendly, and engaging yet secure enough to safeguard user's data.

User Boundaries

Creating an environment where boundaries for users are identified properly is not straightforward and can consume serious resources during development. The role of each user needs to be clarified to make the system more secure. The resources should be distributed accordingly and need to be protected from the outside environment.

Secured Transactions

The database transactions need to be secured enough to not get attacked by malicious injections like SQL injections. The user inputs need to be validated to protect the system from these activities.

Complex Implementations

The implementation of security features is not straightforward and it also requires both financial and human resources. The handling of user credentials can be complex to make it error-free.

Secure Communication

The appropriate protocols like HTTPS are needed to communicate safely between client and server. The data during transactions with the database also needs to be protected.

Third-party Integration

The developer needs to be extra careful when taking services from external systems. This can be the use of Application Programming Interfaces, dependencies, external libraries, and direct integration of a section from another service provider. Regular updates are also needed and numerous conflicts can appear later.

Best Practices

The following methods can be adopted by the developers to ensure that the system is secured for the end users,

- Use secure defaults for better configurations of authentication and authorization.
- Maintain logs to identify any distinct activity and create measures to handle these activities.
- Provide an additional layer of security to avoid common attacking methods.

- Regularly update the system and keep the dependencies updated to ensure secure usage.
- The third-party integrations like libraries, APIs, and other dependencies need to be up-to-date and secured.
- Provide security training to development staff responsible for the implementation to use appropriate coding practices.
- Use regular testing (mostly automated testing) to identify recent attacks. Update the components causing the problem.

Research Impact

The research shows the possible methods that can be adopted to avoid security issues in front-end applications and provide a secure interaction of users with the system. The numerous advantages that can be gained from the industry after implementing the suggested techniques are given below,

- The security breaches into the system can be avoided and only authorized users can gain access to the system. This can be achieved with advanced authentication and authorization methods.
- The regulatory requirements can be maintained by the organization in terms of privacy policy and security.
- The user data becomes more secure and helps gain the trust of the user by providing the integral system.
- The big data of large applications from organizations would be protected and incidents of security attacks can be minimized.

Future Developments

More advanced methods are being developed which will help to secure the front-end applications. The improved authorization controls would assist in preventing unauthorized access and security breakage. The automation tools are under the development phase which will make the application data more secure. Blockchain technology is going to play a vital role above all other practices with its decentralized approach to managing transactions. The security of microservices will help to secure the overall setup. Other than that, the zero-trust security authentication will help in continuous authentication and will work on the geographical location of the user [12].

Conclusion

In the end, the user experience is directly linked with the front end of the application and the security it provides to protect their data. However, using security techniques during implementation while keeping user engagement can be complex sometimes when working on larger systems. It can consume different kinds of resources from the development end but the numbers can be reduced by employing suggested techniques and considering best practices to avoid security issues.

References

[1]. R. C. Ribeiro, E. Canedo, B. J. G. Praciano and G. P. M. Pinheiro, "Front End Application Security: Proposal for a New Approach," in 22nd International Conference on Enterprise Information Systems, Jan. 2020.

[2]. P. Mahindrakar and U. Pujeri, "Insights of JSON Web Token," International Journal of Recent Technology and Engineering (IJRTE), vol. 8, no. 6, Mar. 2020.

[3]. P. Bruegger, "Authentication and Authorization," Advanced Software Engineering Topics: Aspect Oriented Programming, p. 21, 2006.

[4]. P. Cigoj and B. J. Blažič, "An Authentication and Authorization Solution for a Multiplatform Cloud Environment," Information Security Journal: A Global Perspective, vol. 24, no. 4-6, pp. 146-156, Aug. 2015.

[5]. B. Paul, "Authentication and Authorization for the front-end web developer," Aug. 2020.

[6]. S. A. Kumar and Y. U. Rani, "Implementation and analysis of Web application security measures using OWASP Guidelines," in International Conference on Recent Trends in Microelectronics, Automation, Computing and Communications Systems (ICMACC), Hyderabad, India, Dec. 2022.

[7]. S. K. Lala, A. Kumar and S. T., "Secure Web development using OWASP Guidelines," in 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, May. 2021.

[8]. X. Li, "Detection and prevention of logic attacks against web applications through black-box analysis," 2013.

[9]. R. Carbone, L. Compagna, A. Panichella and S. E. Ponta, "Security Threat Identification and Testing," in IEEE 8th International Conference on Software Testing, Verification and Validation (ICST), Graz, Austria, Apr. 2015.

[10]. N. Nagaratnam, A. Nadalin, M. Hondo, M. McIntosh and P. Austel, "Business-driven application security: From modeling to managing secure applications," IBM Systems Journal, vol. 44, no. 4, pp. 847 - 867, 2005.

[11]. Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz and E. Akin, "A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions," Mar. 2023.

[12]. J. G. H. Y. X. Y. Mingyang Xu, "Zero-Trust Security Authentication Based on SPA and Endogenous Security Architecture," Feb. 2023.