International Journal of Advance Engineering and Research Development
Scientific Journal of Impact Factor (SJIF): 4.72
Special Issue SIEICON-2017,April -2017
e-ISSN : 2348-4470 p-ISSN : 2348-6406

Analysis of Security Threats On Data In Distributed Application

Disha Patel[1],Hitarthi Hora[2], Prof. Ajaysinh Rathod[3]

[1.] *CE Department, Vadodara Institute of Engineering, Kotambi, Gujarat*
[2.] *CE Department, Vadodara Institute of Engineering, Kotambi, Gujarat*
[3.] *CE Department, Vadodara Institute of Engineering, Kotambi, Gujarat*

Abstract: *Now a day's security is a priority to pass the client's data in trusted way to the server. The system of analyzing the security threats on data in distributed application is concerned about providing the security threats related solutions to the clients. The purpose behind the idea is to make a file like a .jar file which include the solution related to the security vulnerabilities so that if any web developer wants to create a web application he needs to import the .jar file only to provide server side securities.*

Keywords**:** *security,threats,prevention,vulnerabilities*

1. INTRODUCTION

Security is prior concern for any web-application. Through this API we will be providing security against vulnerabilities like invalidated input, broken access control and broken authentication and session management. To protect any web-application programmer must write a code from both the side that is client as well as server. Here, when programmer applies the code of security at each component or layer the code redundancy can be generated. Also providing security can be considered as a secondary concern while providing the major features in modules. To overcome this problem of code redundancy and providing the security to both client and server, the idea of library file is come. The code for client side can be stored in .js file and for server side it is .jar file. Only by importing the API in the web-application user can protect the application against security threats.

SCOPE: The idea behind the .jar file is for providing security to web-browser and web-server against different web-attacks. In this API, we will try to initially include the code for the following three attacks:

- Unvalidated input
- Broken access control
- Broken authentication and session management

Unvalidated input: System will solve the problem against improper validation and provides a proper validation, integrity and check input as per business rules. It will also solve problem of SQL injection,cross site scripting and path traversal.

Broken access control: System will provide authorization as per user's accessibility. It will solve problems regarding insecure id, file permissions, path traversal, forced browser past access control, client side caching.

Broken authentication and session management: System will provide security check regarding authentication and session management. We will add a secret question for authenticating the user when username and password is entered and code for different constraints for managing the session.

2. SCENARIO OF EXISTING SYSTEM

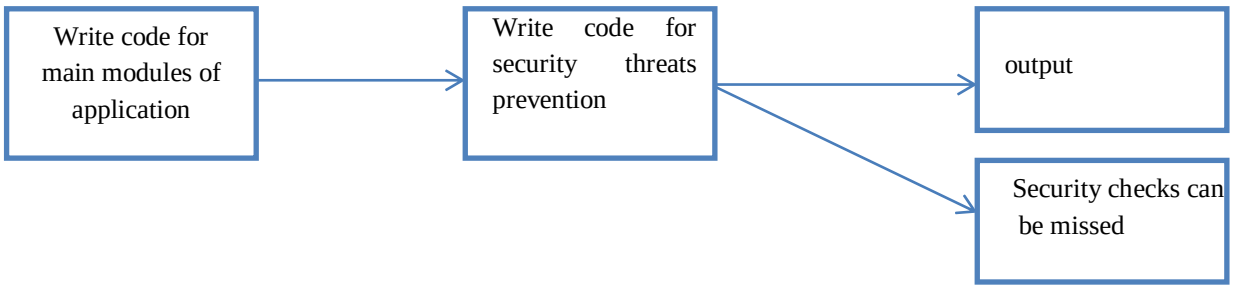
Now-a-days, all the web developers have to mention code to detect and prevent the security related issues. Major issue is that when a web developer is providing thefeatures of application, it might be possible that security checks can be missed and data can be harmed by third party. There are number of ways to harm the server side or client side data so in this case if we miss any security check, client's sensitive information can be stolen.

As per existing scenario there are various methods provided to secure the system but programmer needs to add all the prototypes in every web application and it might be complex to add the security check at each level. There is another drawback that it can increase program length where programmer has to think about not only provide main features of application but also take care of security.

SQL Injection: SQL injections are caused by unchecked user input being passed to a back-end database for execution. When exploited, a SQL injection may cause a variety of consequences from leaking the structure of the back-end database to adding new users in the system [1].

Cross-site Scripting Vulnerabilities: Cross-site scripting occurs when dynamically generated Web pages display input that has not been properly validated. An attacker may embed malicious JavaScript code into dynamically generated pages of trusted sites[1].

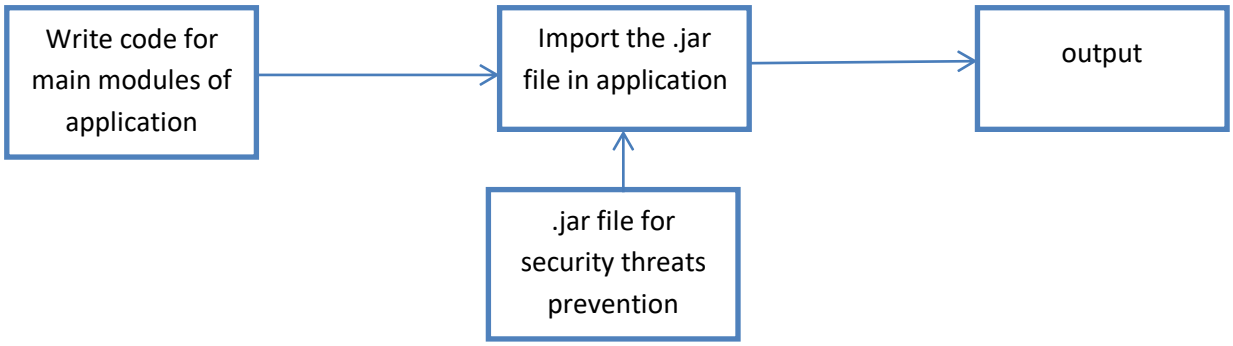
Path Traversal: Path-traversal vulnerabilities allow a hacker to access or control files outside of the intended file access path. Path-traversal attacks are normally carried out via unchecked URL input parameters, cookies, and HTTP request headers[1].



3. SCENARIO OF OUR THEME

The idea behind making the library file which contains security prevention code is to provide the security to data not by detecting them only but prevent the vulnerabilities. We cannot miss any security check even if it has client side that is the html elements or at server side.

One advantage is that web developer should not write the code of security at each level or at each html element instead of this they have to only import the .jar file at the server side so it can be easy for them to provide security to sensitive data and also the chances of miss the security checks is very less as the library contains prevention methods of security vulnerabilities.



4. ALGORITHM

4.1 SQL injection:

Start

Input: get the username/password from user if string

contains '='/'or'/'OR'

then query is invalid else

query is valid end.

4.2 Client Side caching

Start

Input: call method cacheControl()

cacheControl( ):

Step 1: set no-cache in cache-control Step 2:

set no-store in cache-control

end.

4.3 AuthenticationFactor

Start

Input: call method authenticationFactor()
authenticationFactor( ):
Step 1: enter secret question and answer along with username & password Step 2: values matches with database
if(values match)

then user is authenticated

else

user is not authenticated.

End.

5. SUMMARY

We have included three major issues of security that is SQL injection, Client Side caching and authentication factor. In SQL injection, we find the string that contains any value which makes query always true then that query is not valid.
For client side caching, we should not let the browser to store the data into cache when user signs out. We added another field like secret question and answer along with username and password for authentication ofuser.

6.CONCLUSION

So, overall by importing our library containing jar and js file in any web application. we can prevent that website by vulnerabilities mentioned in that.(i.e 1.unvalidated input 2.Broken access control 3.Broken authentication and session management).

7. REFERENCES

[1] V.Benjamin Livshits and Monica S. Lam, "Usenix Security, 2005", Computer Science Department,Stanford University.

[2] http://searchsoftwarequality.techtarget.com/tutorial/Top-10-Web-application-security-vulnerabilities.

