Building Security to REST API

Swetha R
M Tech Scholar
Dept of Information Science and Engineering
R V College of Engineering, Bengaluru

Kavitha S N
Assistant Professor
Dept of Information Science and Engineering
R V College of Engineering, Bengaluru

Abstract:- REST stands for Representational State transfer which is an architectural style designed for web services that are loosely coupled. It is principally used to develop light weight, quick, adaptable and simple to keep up, web services that often utilize HTTP for communication. Industries are enormously dependent and accepting REST based architectures due to its ease of use. The REST web services must be secured from various forms of web attacks due to its light weight nature. Since REST is stateless in nature, the techniques for securing these services are different from standard web application which is managed by session management, but in the case of REST, the calling point may or may not be a web browser, so no session can be maintained. There are several mechanisms to secure REST based web services. In this study the focus is on: HTTP Basic, Digest Authentication, OAuth 2.

I. INTRODUCTION

The enterprises are quickly moving towards the advancement of decentralized, flexible and layered application architecture(E.g. SOA) which can provide solutions to clients on cross platform environments. To make the transition, the treatment of application level of security is considered as a major threat. The security of application has turned out to be critical since the rise of web 2.0.

There has been an explosion of public API ‘s due to the adoption of ease and openness, which enables developers to call the functions and its data from multiple services to create new applications. The Google decoding and Facebook APIs are Representational State Transfer (REST) based web services. As anyone might expect, the application engineers have started to exploit this innovation to encourage business forms in enterprises.

The REST APIs security is still in question and not mature due to following reasons:

- There are nopre-defined security methods for REST architecture. So, developers define their own side of implementation.
- REST API is vulnerable to the same class of web attacks as standard web-based application which include: Injection attacks, Replay attacks, Cross-site scripting, Broken authentication etc.

II. TECHNOLOGY BACKGROUND

Restful applications use HTTP requests to perform CRUD operation such as to post, read, update, and delete data. The web is defined by REST as distributed hypermedia i.e., hyperlinks within hypertext whose resources communicate by exchanging their resource states through representations. REST resembles a three-wheeler that depends on Resources, Representations and Verbs.

A. Resources

The fundamental elements of the web platform are known as Resources. Every resource consists of unique identifier called as universal resource identifier(URI). In REST web services nouns are used to identify a type of resource. For e.g. employee details can be accessed using the URL: http//Emp_service/Emp/1.

B. Verbs

HTTP verbs depict the action needed to be performed on the host. The number of actions client can trigger on the host are:

- GET: retrieve an existing resource
- POST: Creates a new resource
- PUT: modifies an existing resource
- DELETE: removes an existing resource

C. Representations

It is a way in which resources can be showcased to clients. REST supports multiple formats without any restrictions such as: HTML, XML, JSON, Plain Text, GIF, JPEG etc.

III. FRAMEWORK AND IMPLEMENTATION

REST web services are built using Model View Controller (MVC) framework. The model-view-controller architecture and ready components are provided by spring web MVC framework that can be used to develop web applications which are flexible and loosely coupled. The MVC pattern separates the different aspects of the application (input logic, business logic, and UI logic), while providing a loose coupling between these elements.

- The Model consists of POJO in general and encapsulates the application data.
- The View oversees rendering the model data and in general it produces HTML output that is interpreted by client’s browser.
- The Controller is responsible for processing client requests and developing appropriate model and passing it to the view for rendering.

- The Spring Web model-view-controller (MVC) framework is designed around a Dispatcher Servlet that handles all the HTTP requests and responses.

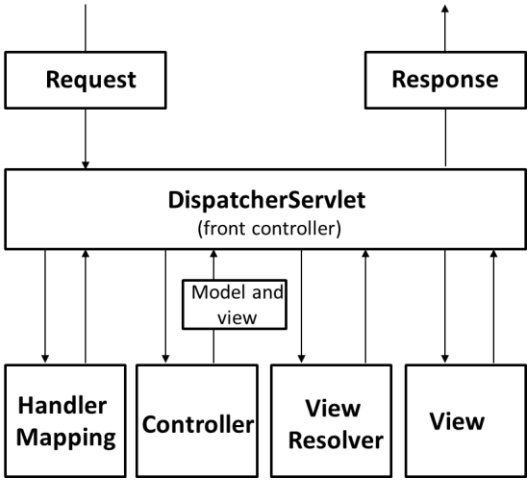

Fig 1:- Spring MVC flow

The figure 1. depicts spring MVC flow where, the Dispatcher Servlet will receive user request and with the help of Handler Mapping it identifies the Controller class name associated with the given request, so that request transfers to the controller.

The Controller will process the request by executing appropriate methods and returns Mode And View object (contains Model data and View name) back to the Dispatcher Servlet. The view (which can be JSP) is then resolved by Front Controller by consulting the View Resolver object.

The selected view is then rendered to the Dispatcher Servlet and back to the client.

Security for REST API can be built in multiple ways such as Basic Authentication, OAuth2 and Digest authentication.

A. Basic Authentication

In Basic Authentication the username and password are base65 encoded. It is stateless in nature as it does not keep track of user sessions and therforebase64 encoded username and password must be sent in each API request. The server adds WWW-Authenticate: Basic realm="messages" to the header of the response where Basic indicates that it is a Basic Authentication, and realm is a string that indicates which part of the site is protected as shown in the Figure 2.

The authorization header is decoded using base64 end in the server end and username and password are extracted. The user can access protected resources if the authentication is successful.

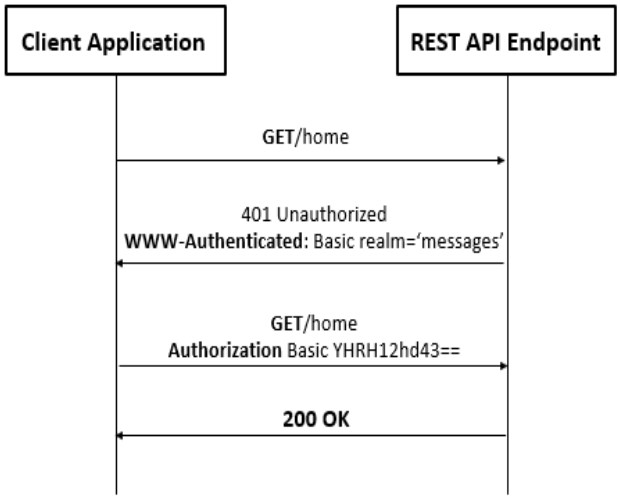

Fig 2:- Basic Authentication

The problem with Basic Authentication is it transmits password as a plain text, though, base64 encoded it can be easily decoded. Therefore, Basic authentication must be used in HTTPS environment only.

B. OAuth2

OAuth2 is a standard authorization protocol which is based on access token concept. It enables applications to communicate with the securely hosted resources in third party services without requiring the resource owners to share their passwords.

The Figure 3 depicts the working of OAuth2 protocol where a user passes credentials to the Authorization server. It authenticates the user and generates an access token with a limited period and a refresh token to the user as a response. The user calls Resource server to access protected resources by passing the access token in the header. The refresh token is used to renew the expired access token.

The Resource server extracts and validates the token with Authorization Server. The user gets access to the protected resources if the authorization is successful.

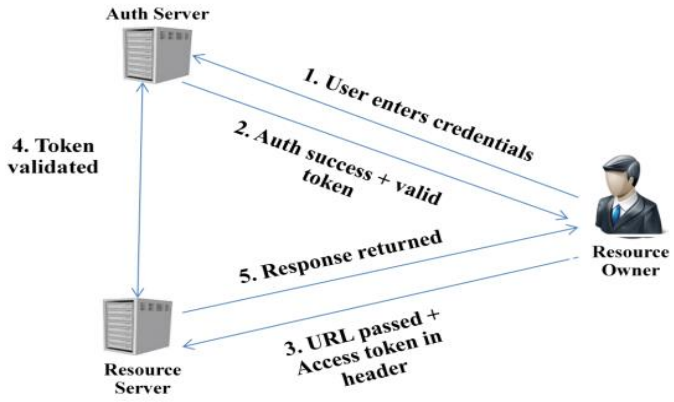

Fig 3:- OAuth2 Protocol

C. Digest Authentication

Digest Authentication is called as challenge-response protocol to authenticate users. It uses MD5 cryptographic hashing algorithm combined with nonce values to hide the password information from replay or malicious attacks.

The Figure 4. depicts the working of Digest Authentication where the client sends a request to the server for protected resources. The server challenges the client to authenticate and sends the required information like realm and nonce to the client. The client calculates the response value and sends it together with a username, realm, URI and nonce to the server. The server computes the hash on its own and compares the two. If they match, it sends the requested datato the client.

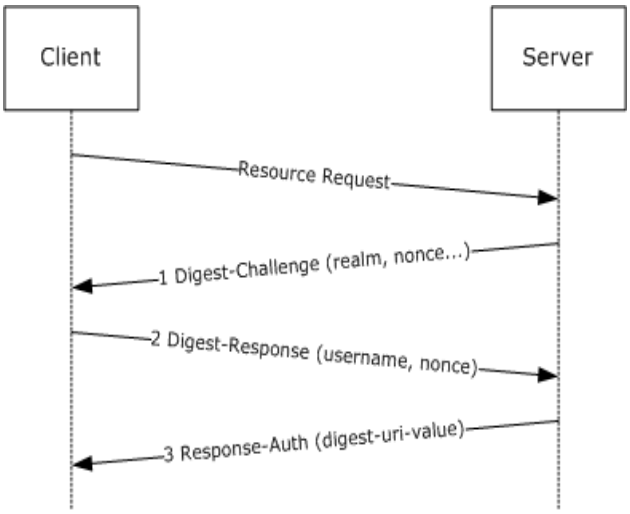

Fig 4:- Digest Authentication

In summary, Digest Authentication does not send passwords in the plain text over the network, due to which replay attacks are prevented and it guards against message tampering.

IV. CONCLUSION

The security for REST is much required due to its flexible nature. Therefore, it validates the user who access the services in various ways. OAuth and Digest authentication are few methods of building security to REST APIs which are more secure than Basic authentication.

The Basic Authentication can be more secure than Digest when combined with SSL.

Currently, developers areusing OAuth protocol to secure most of the REST APIs but, in coming years there will be many other authentication methods with different cryptographic techniques that will strive against security vulnerabilities and attack vectors providing safe and secured REST based APIs.

REFERENCES

[1]. Security provisioning for RESTful web services, Bhumi N. Nakhuva,Tushar A, from Department of Computer Science, L. D. College of Engineering, Ahmedabad, Proceedings of 2017 IEEE conference
[2]. Study on REST API Test Model Supporting Web Service Integration, Hu Wenhui, Huang Yu, Center for Software Eng., Peking Univ., Beijing, China, 2017 IEEE conference
[3]. Nonce-based authenticated key establishment over OAuth 2.0, Renzo E. Navas, Manuel Lagos, 2016 IEEE conference
[4]. Al Shahwan F, and Moessner,K. 2016.Providing SOAP Web Services and Restful Web Services from Mobile Hosts. Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference
[5]. F. Belqasmi, C. Fu, and R. Glitho. 2015.RESTful Web Services for Service Provisioning in Next Generation Networks, A Survey. IEEE Communications Magazine, Vol. no 49, No12, pp.66-73.
[6]. R. Fielding. 2015. Architectural Styles and the Design of Network-based Software Architectures, PhD thesis, University of Califormia, Irvine, USA.
[7]. Ricardo van den Broek. 2014. Comparing the performance of SOAP and REST PHP clients. 14th Twente Student Conference on IT, Enschede, Netherlands.
[8]. Tsenov M. 2013.Web Services Example with PHP/SOAP. International Conference on Computer Systems and Technologies - CompSysTech'2010, Veliko Tarnovo , Bulgaria, pp. IIIA.10-6.