Get started Open in app

Asfiya $ha!kh

Follow 434 Followers About

Web Services & API Pentesting-Part 2

Asfiya $ha!kh Jun 13, 2019 · 5 min read

Hello pentesting rockstars, hope you have skimmed through the part-1 of this blog series.

If not, here is the link.

Let's move forward and have a look at some APIs & Webservices and try to spot the white rabbit manually, later we can check-out the automated tools to find the vulnerabilities.

***Note-** Here I have setup DVWS for the attacks — DVWS is Damn Vulnerable Web Services which is an insecure web application with multiple vulnerable web service components. Also it is a good resource for learning basic API pentesting and can be found here — https://github.com/snoopysecurity/dvws*

We will be learning to exploit the DVWS application for following vulnerabilities.

1. *WSDL Enumeration*
2. *XML Structural Testing*
3. *Malicious SOAP Attachments*
4. *XML External Entity Injection*
5. *XML Bomb Denial-of-Service*

6. *XPATH Injection*
7. *WSDL Scanning*
8. *Cross Site-Tracing*
9. *OS Command Injection*
10. *Server Side Request Forgery*
11. *REST API SQL Injection*
12. *Same Origin Method Execution*
13. *JSON Web Token (JWT) Secret Key Brute Force*
14. *Cross-Origin Resource Sharing*

Huuurrrehh, too much vulnerable app.

Let's understand one by one.

1. WSDL Enumeration

Spider DVWS with burpsuite and then check for services.php folder, there are some requests like check_user_information, return_price, owasp_apitop10 and population etc that can be processed by SOAP web service as shown in the Exhibit.

Load any of the web service response in browser.

*And check if WSDL can be enumerated by appending **?wsdl** after the URL as shown below.*

If only sample request is provided to us with a URL, then check if WSDL can be enumerated by an attacker as shown. Check the WSDL of the web service to find a hidden operation that is not used in a standard SOAP Request and try to invoke them. Also check if web service gives you any confidential information.

2.XML Structural Testing

Malformed xml in SOAP requests may generate unhandled exceptions.

For example, XML parsing exceptions can be thrown by XML parser with overlapping elements, or with open tags that have no matching close tags

A standard XML request would look like this -

Malformed XML structural testing would be to include a duplicate element or an element without closing tag as shown, which can throw an unhandled XML parsing error.

Also, it is possible to perform DOS attack when DOM based parsing is being used as DOM based parser loads everything in the request in server memory. By sending a very large and unexpected payload as shown, DOS can be attained.

DOS can happen easily when blob or exe files are present as binary attachments in web service request. Web service attachments are in base64 encoded format. By attaching a very large base64 string to the message, a hacker may consume parser resources to acquire unavailability of web service.

UNEXPECTED LARGE BLOB

3.Malicious SOAP Attachments

There exists a concept called SOAP with attachments. These are SOAP web services that accepts attachments. These Web Services can be vulnerable if malicious file is possible to upload using them. The danger is in the processing of the attachment on the server and fetching of that file from the server to the users.

An attacker can craft an XML SOAP request containing malware as an attachment and send it to a web service that accepts attachments.This uploaded malware can be executed by an attacker on the server depending on how much vulnerable applications or APIs the server hosts.

If WSDL is provided to us , we need to check for the web service that accepts attachments-

For example:

And then try to attach a test virus attachment such as a non-destructive virus EICAR, to a SOAP Web Service. EICAR is used in below example.

SOAP message with EICAR attachment (as Base64 data):

If web service is not validating the attachment then expected result would be: A SOAP response with the FileUploadResult parameter set to true (this will vary as per service). And the EICAR test virus file is uploaded on the host server and can be redistributed as a PDF.

Now, let's check for SOAP with Attachment vulnerability

The check is similar, however, the request would be similar to the following (note the EICAR base64 information):

If the service is vulnerable then the expected result would be :

The EICAR test virus file is uploaded on the host server and can be redistributed as a TIFF file.

Stay Tuned for the next blog buddy. Happy Hacking :)

References-

https://www.owasp.org/index.php/OWASP_Testing_Guide_v3_Table_of_Contents

https://www.guru99.com/security-web-services.html

Cybersecurity Pentesting API Security Owasp

About Help Legal

Get the Medium app