CHAPTER 6

Four Tier Validation System to prevent Session Hijack attacks by defending the Cross Site Scripting attacks

6.1 INTRODUCTION

Network plays an important role in current scenarios such as online shopping, online ticket booking and online reservation, etc. Even though networks provide lot of flexibility and solutions to the users the numbers of attacks and vulnerabilities are increasing day by day. The users of networks are establishing the web session (http session) with the web server in the application layer.

Web application session involves data exchange between the client and server. Session is classified as stateful session and stateless session. In stateful session, the session information is saved by either client side or server side. In stateless session, the session information is not saved by client and server because of each and every http request is independent.

6.2 SESSION HIJACK ATTACKS

The hackers or attackers make use of the session information to launch the session hijack attacks. Sessions hijack attack is one of the top 10 web attacks as per open web application security project (OWASP) report for years 2012 and 2013. The hacker or attacker hijacks the web application session from the server and interacts with the client in the name of server.

6.3 CROSS SITE SCRIPTING (XSS)

A type of security flaw found in the web application called cross site scripting attacks. Attacker creates and injects the scripts usually written in java called as java scripts to the

web pages of the web application in order overcome the access control policy. Most of the web applications involve with high style Graphical User Interface (GUI) design which needs huge number of java scripts.

Cross site scripting attacks usually occurs in the application layer level and it has the highest impact on security of a web page of the web application. The attacker uses the java script and embeds the java script in to the hyper link of a web page of the particular web application and sends to the client. The client is receiving the hyper link and the client browser executes the hyperlink which consists of hidden java scripts.

The attacker can use and inject the following scripts into the web page of the web application:

(i) Java script
(ii) VB script
(iii) Active X
(iv) HTML tags

The following Fig 6.1 explains the steps involved in executing the cross site scripting attack.

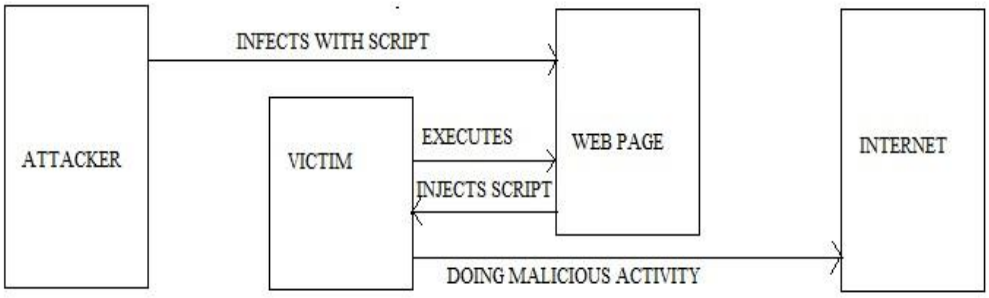

Fig.6.1 Cross Site Scripting attack

Steps involved in executing the cross site scripting attack is given below

(i) Attacker creates the java script.
(ii) Attacker embeds the java script to a web page of the web application.
(iii) Client clicks the hyperlink of that web application.
(iv) Client's web browser executes the java script.
(v) Confidential information is known to the attackers.

6.4 METHOD OF INJECTING THE JAVA SCRIPT

Attackers can use html and java scripts to inject the scripts to the web page. Generally, java script is the broadly used script within dynamic web pages. The method of injecting java script to the web page is called code injection.

Attacker implants the following script labels to inject the scripts into the web page.

<script>
<object>
<applet>
<embed>

Web browser's of the client executes the java script in order to target the information sources such as forums and guest books.

Example of code injection:

<Script>
alert("malicious code injection")
</script>

Attacker can create java scripts by using the following HTML Tags.

<DIV>
<FK>
<IFRAME>

For example, an attacker can use <IFRAME> HTML tag to destroy the client's web browser components such as address bar and bookmarks. Cracking the session ID of web application is done using Java Script. Attacker can use the following java script to create the session ID of a web application.

JavaScript: void(document. Cookie="JSESSION ID")

6.5. TYPES OF XSS ATTACKS

Cross Site Scripting (XSS) attacks can be classified as 3 types. They are as follows

(i) Reflected XSS or non-persistent attack
(ii) Stored XSS or persistent attack
(iii) DOM based XSS attack

6.5.1 REFLECTED XSS OR NON-PERSISTENT ATTACK

If the user inputs are not properly validated, then the code is injected into the Web page. The web page URL is set to the victim by email. The victim clicks the URL and executes the hidden java script which reveals the confidential information to the attackers. This type of XSS is called non-persistent or reflected XSS attacks.

The client side user inputs are reflected at the attacker side and hence called reflected XSS attacks. Fig.6.2 shows the execution of reflected XSS attack.

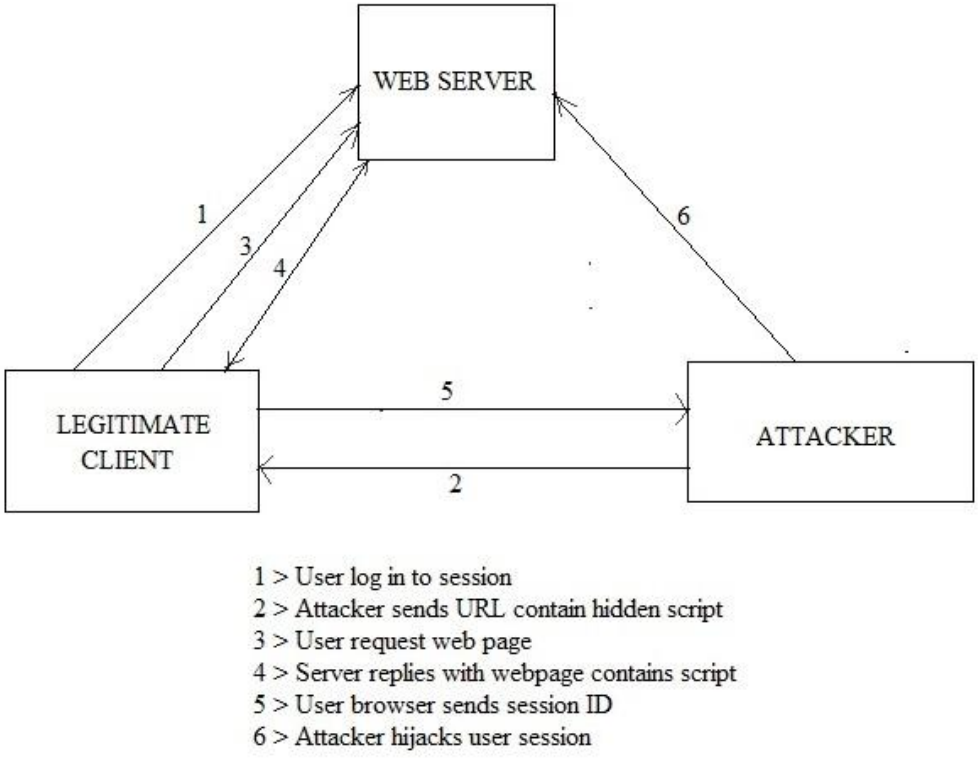

Fig.6.2 Reflected XSS attacks

6.5.2 PERSISTENT OR STORED XSS ATTACK

Persistent XSS attack is an additionally wrecking variation of a cross site scripting attack. The attacker is storing the malicious information to the web server of the web application. An excellent example for this stored XSS attack is with online message sheets where the legitimate clients are permitted to put organized message for different clients to read.

Persistent or type 2 XSS attacks persist most potential vulnerabilities. Type 2 XSS attacks exist when information gave to a web application by a client is initially put away on the server in the database file system or other area. In the long run, this will be shown to the clients in a website page without being encoded utilizing HTML elements. The following Fig.6.3 shows the execution of stored XSS attack.

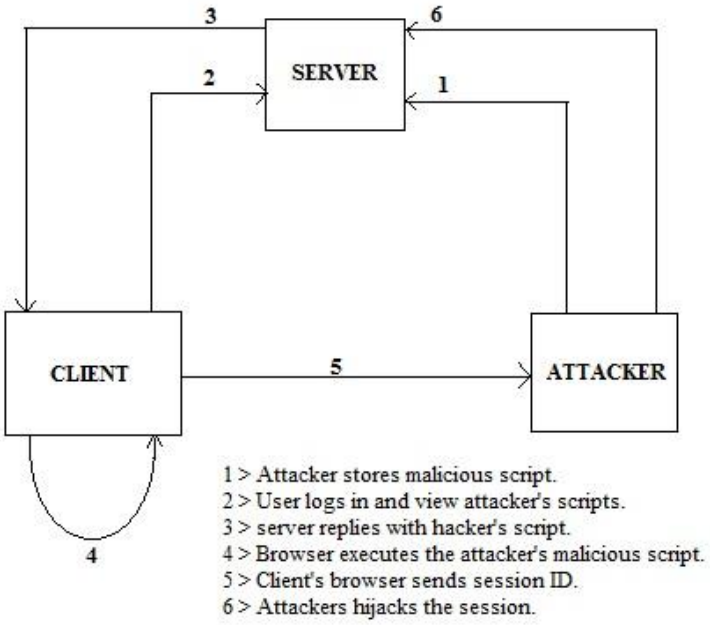

Fig.6.3 Execution of Stored XSS attacks

6.5.3 DOM BASED XSS ATTACKS

Document Object Model (DOM) is a tradition for speaking to and working with articles in a HTML record. Fundamentally all HTML reports have a related DOM comprising of items speaking to the archive properties from the perspective of the program. If any script executed at customer side, the program furnishes the code with the DOM of the HTML page where the script runs.

DOM based XSS is a category of Cross Site Scripting attack which depends on improper taking care of information from its related DOM of the HTML web page. The attacker can use the properties of the HTML document such as URL, location and objects.

6.5.3.1 EXAMPLE OF DOM BASED XSS ATTACK

The client's program gets the compromised URL and send the HTTP appeal to http://www.dombasedxss.com by accepting the static HTML page portrayed previously.

At that point, the program begins constructing the Document Object Model (DOM) of the page with the URL consisting of malignant script. At that point, when the program lands to the script which obtains the client name from the URL by referring the document URL property. The following Fig.6.4 explains the execution of DOM based XSS attacks.

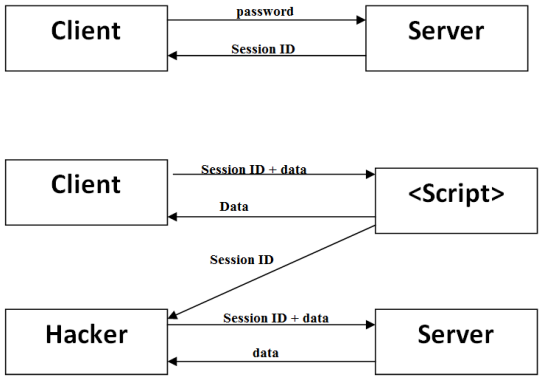

Fig.6.4 Execution of DOM based XSS

6.6 ARCHITECTURE OF A WEB APPLICATION

6.6.1 ACCESSING THE WEB SITE

The following Fig.6.5 explains the basic architecture of the web application. Web browser gets the required data from the database through the web server.

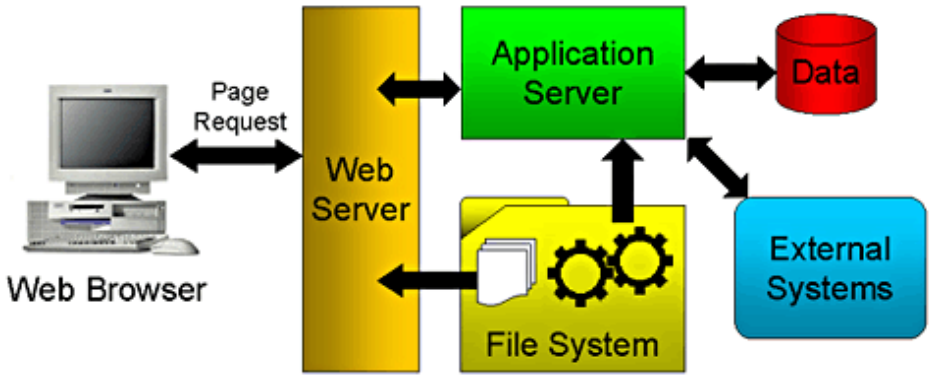

Fig.6.5 Architecture of web application

6.6.2 ARCHITECTURE OF ACCESSING THE WEBAPP SESSION

The general architecture of accessing the web application and the list of steps involved between the web server, client's browser and the client is shown in the Fig.6.6. The Steps of Accessing the Web application session is given below

(i) Registration of new user

(ii) Client login to the web session

(iii)Client's browser stores the session cookies sent by the web server

(iv)Client's http request for web page

(v) HTTP response from the server

(vi)Session data transfer between client and web server

(vii)Log off the web application session

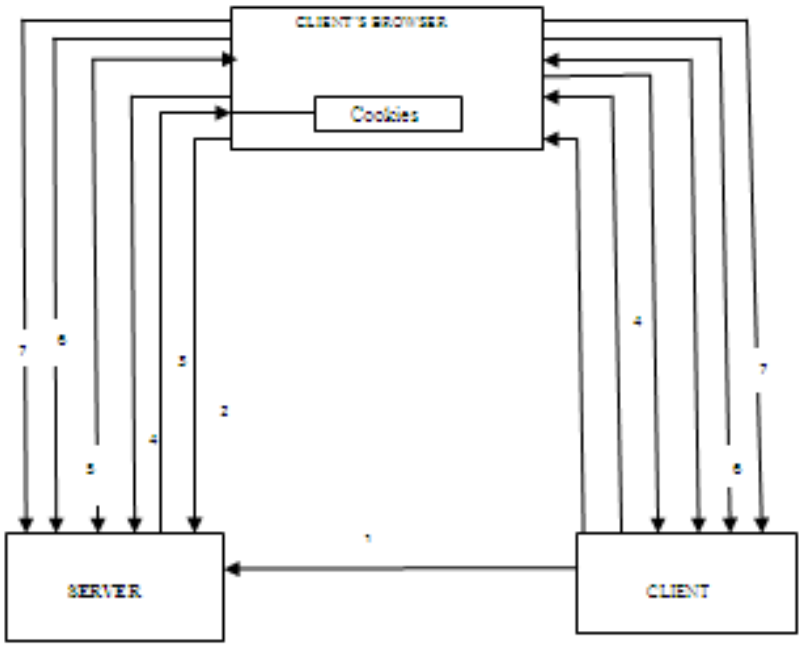

Fig.6.6 Architecture of accessing the web application session

6.7 PROPOSED FOUR TIER VALIDATION SYSTEM

The four tier architecture is proposed to prevent the session hijack attack by defending the cross site scripting attack is shown in the Fig.6.7.

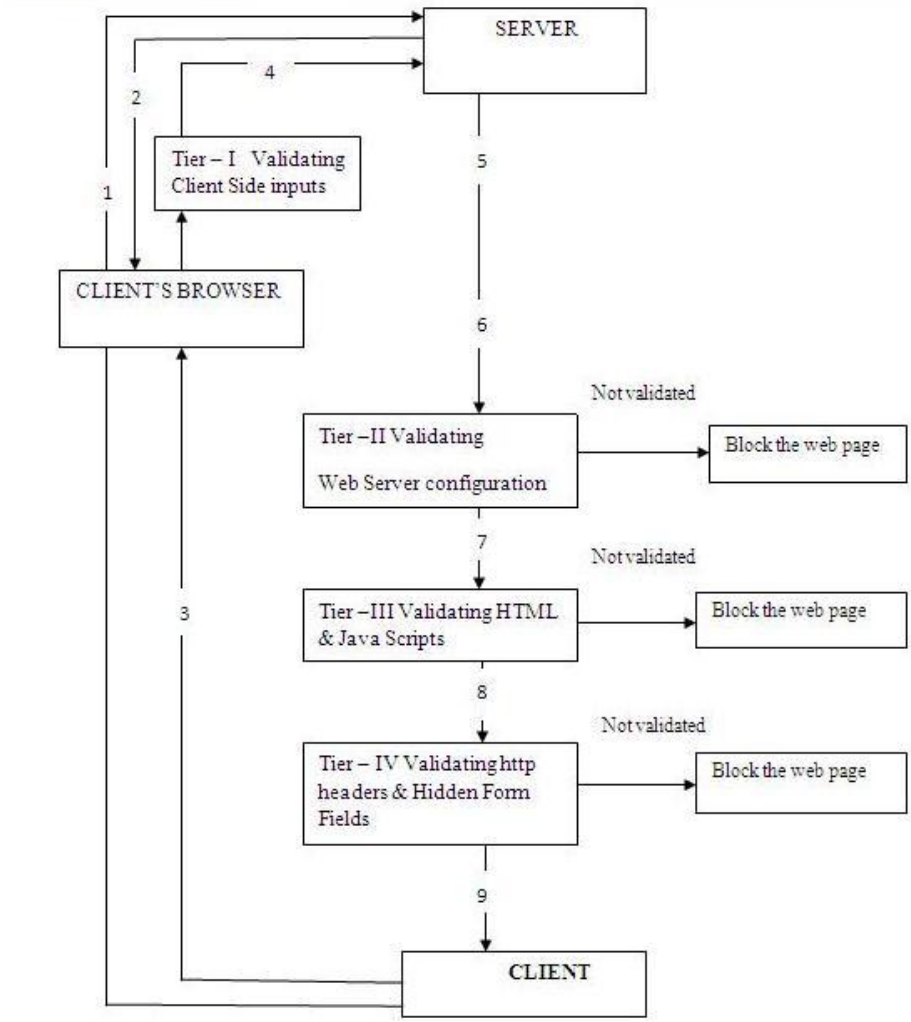

Fig.6.7 Proposed Four Tier Validation System

The detailed steps of the proposed approach and the sequence of web application access steps are described**.** Steps of Proposed Four Tier Architecture

(i) Client login to the web server via browser

(ii) Server send the session cookies and the client's browser stores the cookies

(iii) Client's http request for web application

(iv) Tier - I validation of client side inputs

(v) http response from the web server to client

(vi) Tier II Validation of web server configuration

(vii) Tier III Validation of HTML scripts and Java Scripts

(viii) Tier IV Validation of http header and hidden form fields

(ix) Client receives the web page after the validation of four tiers

6.8 TIER I – VALIDATING CLIENT INPUTS TO SERVER

The client requests the server for web application through http protocol. It is important to check and validate the client side inputs. Especially, the URLs specified by the client should be validated in order to prevent the cross site scripting attacks. The method of validating the client side inputs is shown in the following Fig.6.8. Tier- I validation involves in checking the number of characters in the URL of the web page and validating the various Query parameters available in the URL.

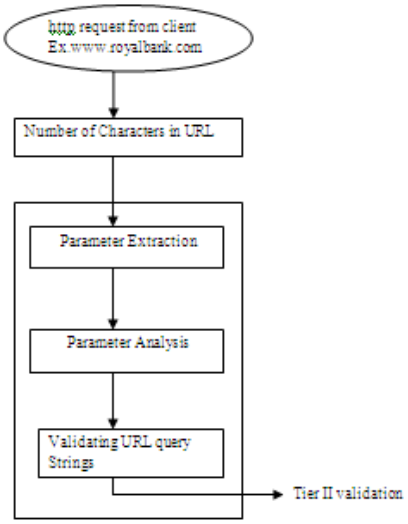

Fig.6.8 Validation of Client inputs

6.8.1 NUMBER OF CHARACTERS IN THE URL

The client places the http request to the web server through the browser. It is essential to check and validate the length of the URL specified in the browser. The maximum length of the URL of standard browsers are given in the Table.6.1

Table.6.1 Allowed length of URL in Browsers

No	Browser	Maximum length of URL
1	Internet Explorer	2,083 characters
2	Safari	80,000 characters
3	Firefox	65,536 characters
4	Opera	1,90,000 characters
5	Search Engines displays URL links up to	2047 characters

If the number of characters in the URL is exceeds beyond the maximum allowed length of the URL of the concerned browser, then the request is denied by the server. Because, the exceeded length of URLs will leads to buffer overflow attack and stack overflow attacks.

6.8.2 VALIDATION OF URL QUERY PARAMETERS

The parameters of the URL queries are extracted from the URL and the parameters are analyzed to allow only the legitimate URLs. jQuery URL Parser is used to extract the parameters in the URL .

The following syntax is used to extract the parameters when the client requests for www.gmail.com

```
$.url().param('
https://accounts.google.com/ServiceLogin?service=mail&passive=true&rm=false&conti
nue=https://mail.google.com/mail/&ss=1&scc=1&ltmpl=default&ltmplcache=2);
```

The parameters such as mail, service, login, true, false and default are extracted from the above URL. The following client side inputs have to be validated to prevent the session hijack attacks.

i) Remove the harmful characters in the URL ; . &, >, < , |
ii) Allow the harmless characters in the URL a….z, A….Z, 0-9
iii) Block the user input that contains Java Script
iv) Block the client input that contains concatenated strings
v) Allow only the parameterized URL query that contains bind value
vi) Run the URL query with minimal privileges
vii) Only 22.3 % web sites out of top one million web sites use http only cookies. So allow the websites that use http only cookies
viii) Allow only standard domain names such as .com, .edu, ..org, etc...

6.9 TIER II VALIDATING WEB SERVER CONFIGURATION

Most of the Web applications are developed using PHP, ASP.NET, and JAVA. The following Table.6.2 shows the % of web applications that use PHP, ASP.Net and Java

6.9.1 WEB APPLICATION PLATFORM

Most of the web applications are using PHP, ASP.NET platforms which is shown in Table.6.2

Table.6.2 Web applications and its platform

No	Programming Language	% of web applications
1	PHP	63 %
2	ASP.NET	19 %
3	JAVA	14 %
4	Others	4 %

6.9.2 WEB SEREVR USAGE

The percentage of web applications that use different web server is listed in the Table.6.3

Table.6.3 Web Servers percentage

No	Web Server	Proportion
1	APACHE	57 %
2	IIS	17 %
3	NGINX	10 %
4	Others	16 %

Table.6.3 shows that 74 % of web applications use Apache and IIS server. It is important to validate the configuration of web server of the web application.

6.9.2 WEB SEREVR CONFIGURATION

Web server configuration is validated. They are listed below.

i) Check the initial settings and configuration of web.config file present in the server. It has the information about the execution of web application.

ii) Separation of web server components

iii) Validating the code libraries

iv) Validating the expired Secure Socket Layer (SSL) certificates

v) Denying the web sites that host Java Script on third party server

vi) Detecting the modified HTML/Java/ PHP files in web applications

vii) Detecting the missing patches in the web server

viii) Detecting the spikes in web error logs that indicate web application probing

6.10 TIER –III VALIDATION OF HTML & JAVA SCRIPTS

Some of the Java scripts and HTML scripts are used to inject the malicious scripts in to the web applications. The attacker uses the following method to steal the cookies from the web applications.

javascript.alert(document.cookie);

javascript:void(document.cookie=”PHPSID=<<another Session ID>>”);

javascript:void(document.cookie=”logged=yes”);

The following tags are used to inject the vulnerability in to the web applications.

<FK STYLE=”behavior:url=http://www.abcbank.com>

<FK STYLE=”behavior:url=http://www.xyzconsultancy.com>

<DIV STYLE=’img:url(javascript:alert(“Injected”)>

The following Fig.6.9 explains the method of separating the HTML and Java scripts from the web pages using soot tool.

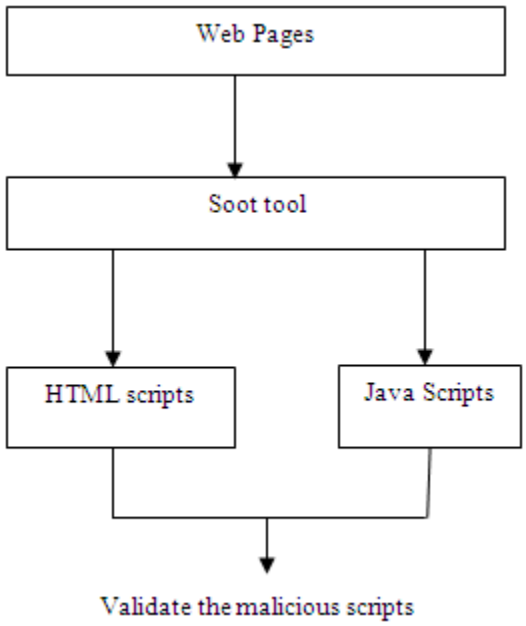

Fig.6.9 Validation of HTML & Java Scripts

Soot tool is used to parse the web pages of web applications in to HTML scripts and Java scripts. A soot tool is the open source tool that is available in the web. The web page is blocked in the Tier III if the malicious java scripts such as Embed, Script are detected in the web page. If the malicious java and HTML scripts are not detected, then the web pages are forwarded to Tier-IV validation.

6.11 TIER –IV VALIDATION OF HIDDEN FORM FIELDS

In tier IV, http response headers and hidden form fields are validated. JQuery Validation is used to validate the hidden form fields of the web application. JQuery is a Quick, little and gimmick rich Java script library. JQuery can be used as traversing HTML documents, handling of events. JQuery has a API which is used among various types of standard browsers.

Most of the web applications have the hidden form fields to display the error message or thank you message to the client. These hidden form fields has to be validated carefully. The following hidden form field displays the error message to the client.

Example:

```
$name = $_REQUEST['name'];

$university = $_REQUEST['university'];

$phone= $_REQUEST['phone'];

$hiddenField = 'You must fill out the Name & university section';
```

The attacker makes use of the hidden form fields to launch the cross site scripting attacks. The hidden form fields have to be validated to prevent the XSS attacks.

6.12 SUMMARY

Current web applications are weakly secured against session hijacking attacks that are executed via cross site scripting attacks. The proposed four tier validation system is used to prevent the session hijack attacks in web applications by defending the Cross Site Scripting attacks. Client side inputs are validated in the first tier and the web server configurations are validated in the second tier. In the third tier, malicious HTML scripts and Java scripts are validated. In the fourth tier, http headers and hidden form fields are validated. The proposed four tier validation system completely eliminates the cross site scripting attack and thus prevents the Session Hijack attacks.