A Secure Web Application: E-Tracking System

Vina M. Lomte [1] , Prof. D. R. Ingle [2,] Prof. B. B. Meshram[3]

[1]Department of Computer Engineering, MGMCET, Kamothe,Navi Mumbai
vinamlomte@gmail.com
[2] Department of Computer Engineering, Bharrati Vidyapeeth, Kharghar,Navi Mumbai
dringleus@yahoo.com
[3] Department of Computer Engineering, VJIT, Mumbai

Abstract

The World Wide Web has experienced remarkable growth in recent years hence security is becoming one of the major promising task in the present scenario of e-business environment. Web attacks can devastate the system within no time. More than 80% attacks are at application layer and almost 90% applications are vulnerable to these attacks. Traditional solution is not capable to protect the web from such attacks.This paper handles different web attacks and also provide some tricks used by hackers to hack the web world similarly it contains an attempt has been made to analyze impact of DOS, SQL injection, Cross site scripting, Sniffing/ Request Encoding on web application in terms of throughput and response time etc. It also provides the best protection mechanisms for the said attacks. Our main aim is to analyze both E-application one with security (proposed E –Tracking system) and another without security and find the impact of all above attacks on both in terms of request time, response time & throughput etc.

.

Keywords

IDS - Intrusion detection system, IDPS- Intrusion detection Prevention system XSS – Cross site scripting, SQL-Sequential query language, DOS- Denial of Services

I Introduction

Now a days web security is biggest issue in the corporate world. The world is highly dependent on the Internet .It is considered as main infrastructure of the global information society. Hence internet plays very vital role in the socio-economic growth of the computer society. Availability of internet and its services makes you available information, security controls in the accessible and operationable form.[1] XSS , SQL injection, Sniffing, Request Encoding and DOS attacks which poses an immense threat to the availability of the Internet. An occurrence of these serious web attacks causes disturbance in the service, communication and resources of the target users. It degrades the web performance. Nowadays to achieve security of distributed systems is a dominant task for any organization including the most modest types of e-commerce, banks and even large state systems However, the increasing number and a variety of system attacks suggest, between among other things, that the design and realization of these systems are often very poor as far as security is concerned. Web security is essential part of business world. [2] Dos attack is an attempt to make machine or network resources unavailable to its intended users .Request Encoding Attack (XSS) is security exploit in which the hacker or attacker can insert malicious coding into a link that appears to be trustworthy source. it causes for execution of injected or malicious code on the victim's machine by exploiting security by validating data .SQL Injection is type of exploit in which hacker can add SQL code to a web from input box or malicious database statements insertion into the textbox as its input to gain the access to resources. Sniffing

DOI:10.5121/iju.2012.3401

(Request Encoding) attack is responsible for data hacking during data transmission. Previous approaches to identifying these kinds of attacks and preventing them includes defensive coding, static analysis, dynamic monitoring, and test generation. [6] These techniques have their own merits but have some drawback like Defensive coding is error-prone and requires rewriting existing software to use safe libraries. Static analysis tools can produce false warnings and do not create concrete examples of inputs that exploit the vulnerabilities [8]. Dynamic monitoring tools incur runtime overhead on the running application and do not detect vulnerabilities until the code has been deployed. Black-box test generation does not take advantage of the application’s internals, while previous white box techniques have not been shown to discover unknown vulnerabilities [10].traditional solution for DOS protecting the network connection's confidentiality and integrity, protecting the server from break-in, and protecting the client's private information from unintended disclosure. A lot of protocols and mechanisms [12][have been developed that address these issues individually. It can take many forms depending on the resources the attacker is trying to exhaust. Because of these attacks Vulnerabilities business market will get hampered and it is headache to the E- business system.[15]

This paper will provide the security solution for each stage of web life cycle. Task involved best practices during information gathering designing, coding and testing with security. Our primary goal to produce secure environment to the business market. Our proposed solution provides different protection mechanisms for DOS, SQL injection, Cross site scripting, Sniffing/ Request Encoding.

II Literature Review

Traditional solutions like Web vulnerability scanner, Web application firewalls, Vulnerability patching, Application-Layer Filtering, Network Firewalls, Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS) are not capable to find application level web attacks like SQL Injection , Denial of services(DOS),Cross site scripting(XSS) and Sniffing/ Request Encoding Attacks. Traditional security measures are reactive in nature and they fail to provide sufficient protection of web world. Organizations are unaware of IT and Cyber risks. An organization is not expense to unnecessary IT and Cyber risks they don’t want to leave profit opportunity. They are based on the assumption that all vulnerabilities and attack methods are disclosed to the public.

- **Web Vulnerability Scanner**

It is computer program that perform the diagnosis of a vulnerability analysis. This process is proactive .It determines if and where a system can be exploited and / or threatened for weakness. It typically returns to the scanning of system that are connected to the internet but can also refer to system audits on internal network that are not connected to the internet in order to assess the threat of rogue software.

A combined solution dramatically lowers the risk of application vulnerabilities in deployments, and it delivers cost savings on vulnerability repairs As many as 70% of web sites have vulnerabilities that could lead to the theft of sensitive corporate data such as credit card information and customer lists. Hackers are concentrating their efforts on web-based applications - shopping carts, forms, login pages, dynamic content,

- **Web application firewalls**

Web application firewall to protect against known attack method but they've become a necessity for any organization that provides Web access to its applications. Web application firewalls are software or hardware devices which are placed to monitor web traffic which is mainly designed to inspect data (header, URL parameter and web content. It has better filtering capabilities. Another drawback of web application firewall is protocol like HTTP, SMTP etc requires its own proxy application and support for new network application. It typically require clients on the network to install specialized software or make configurations changes in the proxy connections and most important issues is proxy security.

- **Application-level gateways**

It is one of the firewall systems. It provides different process which maintains complete TCP connection state and sequencing. It is supported to user level authentication and able to analysis inside the payload position of the data packets, but it is not very much supporting in filtering process hence it slower than packet filters. It should know about internal clients and it is not possible every time connection can be supported.

- **Antivirus**

It is very effective in preventing, controlling and detecting malware, virus and no of potentially hazardous computer program which can damage computer components. It keep increasing day by day hence antivirus software needs to be update periodically. It very useful in preventing signature is known antivirus needs to be updated as soon as a new update is available from the software vendor. It may have high positive false rate. It is not capable in finding unknown attacks.

- **Vulnerability patching**

Patching is a reactive process which only protect web from known vulnerabilities. When the vulnerabilities discovered or disclosed from at that time patch is developed. This is known as "window of exposure. Any hacker or attacker with the prior knowledge of the vulnerability or having a working exploit can gain unauthorized access to vulnerable systems and data. Vulnerabilities can be exploited either by worms, viruses or through a targeted attack. With patching it is not possible to protect the web application from no of web attacks.

- **Network Firewalls**

Traditional firewalls are not capable and sufficient to protect web from web application attacks. They are capable to perfect only there network components are insufficient to protect web application. They only monitor network traffic and block packets based on internet add and port no. Here web application firewall ae used to protect against known attacks signature knowledge base should be available is used for server plug-in or filter that applies a set of rates to an HTTP conversation.

Drawback:

It is network device cost and performance must be 7 Layer OSI, CPU cycles reading and interpreting each packet. Consumes packet filtering and stateful packet inspection look traffic at network layer require more processing power, bottleneck for the network.

Application firewalls are more susceptible to DOS attack and therefore are less suited to high band width or real time application. It is also be Vulnerable to the security to the security loopholes.

- **Intrusion Detection Systems (IDS)- Host-Based IDS & Network-Based IDS**

IDS is truly reactive which monitor network traffic and generate an alert when potentially malicious traffic is detected which helps you to determine what actually happened. IDS help you determine what happened. The main aim of IDs is analyzing or monitoring network traffic based on attacks signature, heuristic method etc for malicious activity if found it will generate alternate to the administration. The main disadvantage of IDs is maintenance and deployment overhead. It is responsible for generating high rate of false alarm. Implementation of IDs very complex in large network environments. There just isn't a single IDS model that offers 100% intrusion detection with a 0% false alarm rate that can be applied in today's complex networking environment.

- **Intrusion Prevention Systems (IPS)**

These are network security appliances that monitor network or system activities for malicious activity. It attempts to block / stop activities and report the same. It should monitor computer network for abnormal activity quite expensive. If there are multiple IPs on the network then every packet of comes loss of network performance and thus also canes another problem rate of false negative is high. Network being slowed down by the IPs these should be separate design for application level IPs. IPS is thus vulnerable to zero-day attacks exploiting undisclosed vulnerabilities

III Proposed System

Proposed Model/Solution (E-Tracking System)
E-Tracking System is a secured web application which contains mitigation for different web application layer attacks like Cross Site Scripting attack (XSS), SQL Injection, DOS, Request Encoding. Our proposed system will provide good protection for all above attacks and after applying proposed protection mechanism to the web application we can maintain high level security.

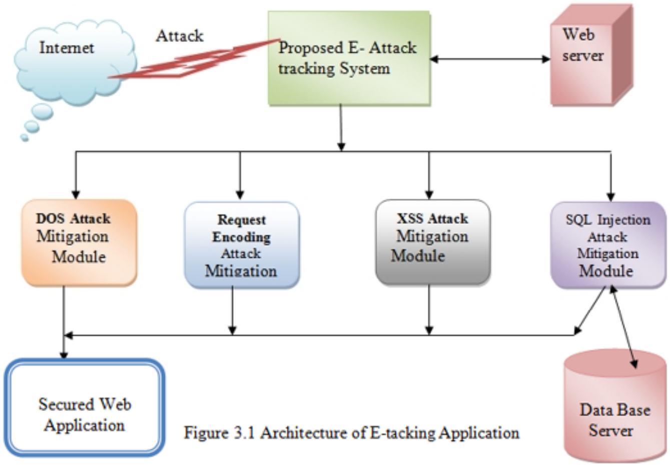

Internet
Attack
Proposed E- Attack tracking System
Web server
DOS Attack Mitigation Module
Request Encoding Attack Mitigation
XSS Attack Mitigation Module
SQL Injection Attack Mitigation Module
Secured Web Application
Data Base Server

Figure 3.1 Architecture of E-tacking Application

3.1 SQL Injection Attack

SQL Injection is the attack technique often attacks the website. This is done by inserting SQL statement in a web form entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database. it is a code injection technique that exploits security vulnerability. This type of vulnerability happens when user input is either incorrectly filtered for string embedded in SQL statements or it is not strongly typed and unexpected execution of the same. SQL commands are thus injected from the web form into the database of an application (like queries) to change the database content or dump the database information like credit card or passwords to the attacker. It is the approach of gaining access to private information is known as SQL Injection. Attacker can try to access sensitive information from the database by firing SQL query that will cause the database parser to malfunction. This type of vulnerability is harmful in financial transactions, educational endeavors, and countless other activities like transferring a balance from a bank account, always comes with a security risk.

Solution:

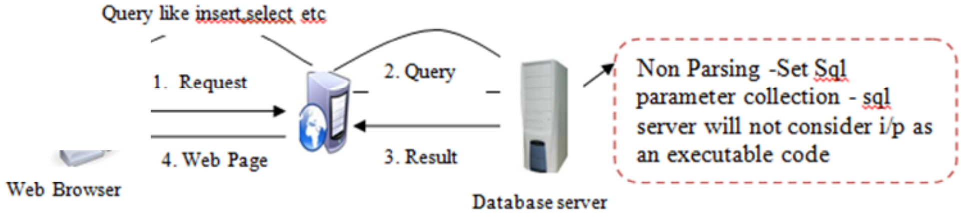

Query like insert,select etc
1. Request
4. Web Page
Web Browser
2. Query
3. Result
Database server
Non Parsing -Set Sql parameter collection - sql server will not consider i/p as an executable code

Figure 3.2 Mitigation of SQL Injection Attack

This attack is very dangerous for E-Business Environment. As mitigation for protecting the web from this attack we can use SQL parameters with stored procedures or dynamically constructed SQL command strings. With the help of type checking we can check un trusted data entered by user. you need to: Constrain and sanitize input data. Check for known good data by validating for type, length, format, and range. **SqlParameterCollection** provide type checking and length validation. It will treat input as a literal value, and SQL Server does not treat it as executable code. An additional benefit of using a parameters collection is that we can enforce type and length checks.

SqlParameter userIdParam = new SqlParameter("UserId", SqlDbType.NVarChar,50); SqlParameter firstNameParam = new SqlParameter("FNAME", SqlDbType.NVarChar, 50); SqlParameter lastNameParam = new SqlParameter("LNAME", SqlDbType.NVarChar,50);

3.2 Denial of Service (DOS) Attack

It is an incident in which a user or organization is deprived of the services of a resource they would normally expect to have. This will typically happen through one of the following ways:

- Crashing the target host system.
- Disabling communication between systems.
- In Intent to make system or network down or have it operate at a lower speed to reduced its performance.
- Lock the system, so that there is provision to automatic rebooting of the same. , so that, production is disrupted. Depending on the thoughts of attacker DoS attacks planned, the attacker first need to plan numbers of computers he wish to make vulnerable.

  Next, the hacker establishes a communication channels between computers, so that they can be controlled and engaged in a coordinated manner

Solution:

1. To handle this attack, each individual request is tracked on custom DB by using **DOSAttack module**.
2. Application determines the threshold value for the time between consecutive requests.
3. Application verifies the timing between the two requests.
4. If the next request is coming from same client within the time less than threshold value then system picks up the Ip address of that client and stores that Ip address into database in table ‘blocked IP’.
5. When the next request receives from the same client then the system checks if this IP address exists in the blocked IP table. If it exists then DOSAttackmodule blocks this request and dosn’t send that request to process further.

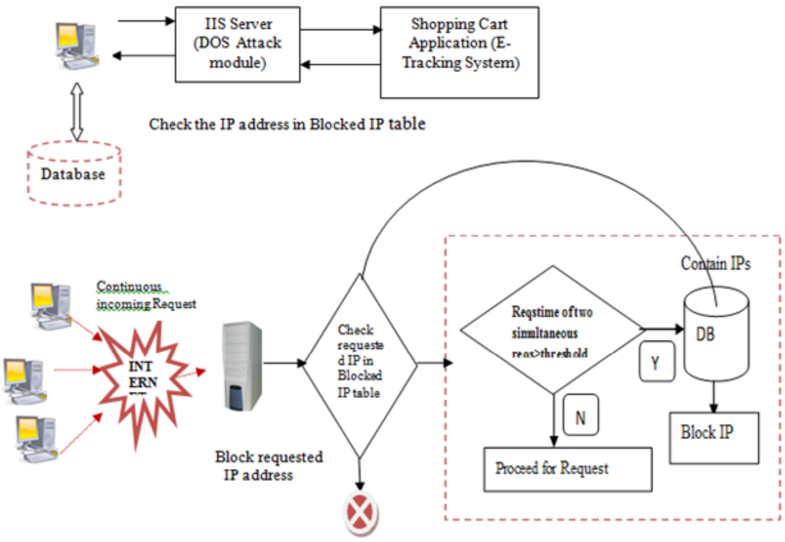

IIS Server (DOS Attack module)
Shopping Cart Application (E-Tracking System)
Check the IP address in Blocked IP table
Database
Continuous incoming Request
INT ERN
Check requeste d IP in Blocked IP table
Block requested IP address
Reqstime of two simultaneous reqs>threshold
Contain IPs
DB
Y
N
Block IP
Proceed for Request

Figure 3.3 Mitigation of DOS Attack

```
string ip = HttpContext.Current.Request.UserHostAddress;
        _Banned =OnlineShopping.modCommon.GetBlockedIpList();
        if (_Banned.Contains(ip))
        {
           HttpContext.Current.Response.StatusCode = 403;
           HttpContext.Current.Response.Write("ERROR: Access Denied");
        }
```

3.3 Cross Site Scripting Attack (XSS)

It is common web application attack commonly attacks by embedding scripts in a page which are executed on the client-side in the user's web browser rather than on the server-side. This attack can happens because of security weaknesses of client-side scripting languages, with HTML and JavaScript (others being VBScript, ActiveX, HTML, or Flash) as the prime culprits for this exploit. It is use to manipulate client-side scripts of a web application to execute in the manner desired by the malicious user. It is code injection which occur when an attacker uses a web application to send malicious code, generally in the form of a browse.

Solution:

We are trying to produce one of effective solution which protect the web application from the XSS attack is telling the browser that the data what you are sending should treated as data and should not be interpreted in any other way. If an attacker manages to put a script on your page, the victim will not be affected because the browser will not execute the script if it is properly escaped .To implement this we have used the **HttpUtility.HtmlEncode** method to encode output if it contains input from the user or from other sources such as databases.

HtmlEncode is very effective utility or method of HTML which is used to replace characters that have special meaning in HTML-to-HTML variables that represent those characters. **Html Encoding** is the action of encoding certain html characters into html entities. Hence the script from the attacker is encoded and treated as a data instead of the script. So the script from the attacker is not executed on the browser and the user is not affected by the XSS attack.

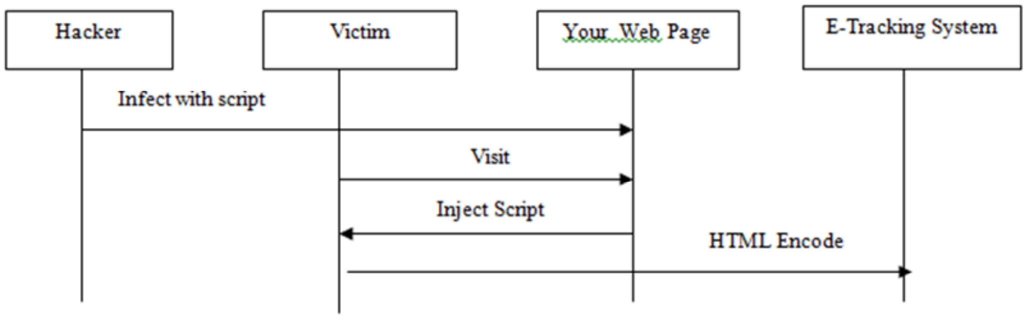


Figure 3.4 Mitigation of Cross Site Scripting Attack (XSS)

```
sql = "INSERT INTO USERDETAILS (USERID, FNAME, LNAME, ADD1, ADD2, CITY,
STATE, ZIP, PHONE, EMAIL) VALUES('" + HttpUtility.HtmlEncode(this.UserId) + "','" +
HttpUtility.HtmlEncode(this.FirstName) + "','" + HttpUtility.HtmlEncode(this.LastName) + "','"
+ "');";
cmd = new SqlCommand(sql, Common.dbConnection);
cmd.ExecuteNonQuery();
```

3.4 Request Encoding

In this type of attack, the attacker tries to decode the request which is traversed between client and server.

After decoding the request he may track the sensitive data from the application.

Solution:

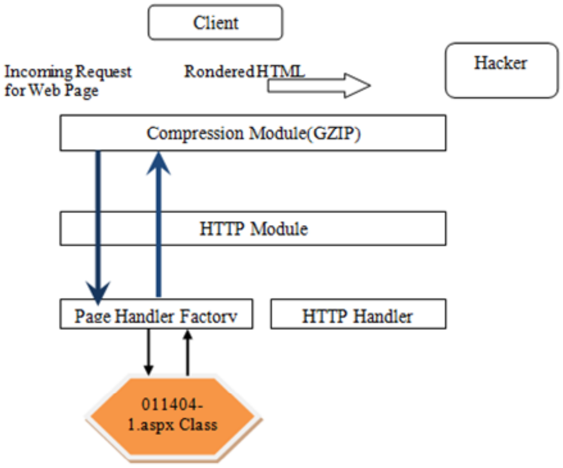


Figure 3.5 Mitigation of Request Encoding

To prevent this attack, we compress and encode the request before sending to the server by using the compression module. When the request is compressed, it can’t be decoded by the hacker in between.

```
app.Response.Filter = new GZipStream(app.Response.Filter, CompressionMode.Compress);
 SetEncoding(GZIP);

  private void SetEncoding(string encoding)
    {
       HttpContext.Current.Response.AppendHeader("Content-encoding", encoding);
    }
```

IV Design Implementation

The architecture of the system is based on Two-Tier architecture

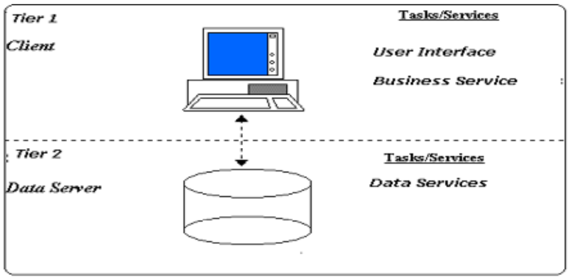


Figure 4.1 Solution Architecture

4.1 Tier 1:

a.User Interface layer:

i. This layer basically displays the data to the user.
ii. This layer consists of all the aspx pages which present the data.

b. Business Services:

i. This layer performs different business operations on the data coming from data access layer.
ii. When specific operation is performed, the data is passed on the User Interface layer to display it.
iii. Similarly, it takes the data from User Interface layer, performs the operations and passes on the data to data access layer to store into Data base.
iv. In the application, the .cs files play the role of business logic layer as they process the data coming from aspx page as well as from database layer.

4.2 Tier 2:

a. **Data Access Layer:**

i This layer contains the class which performs the database operations like fetching the data from database and saving the data into database.

ii. After retrieving the data from database, it passes on the data to business logic layer for further processing.

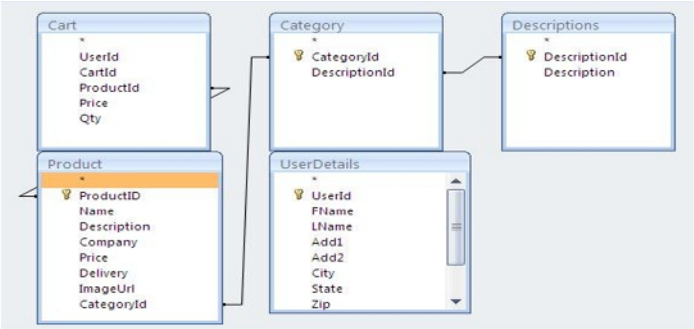
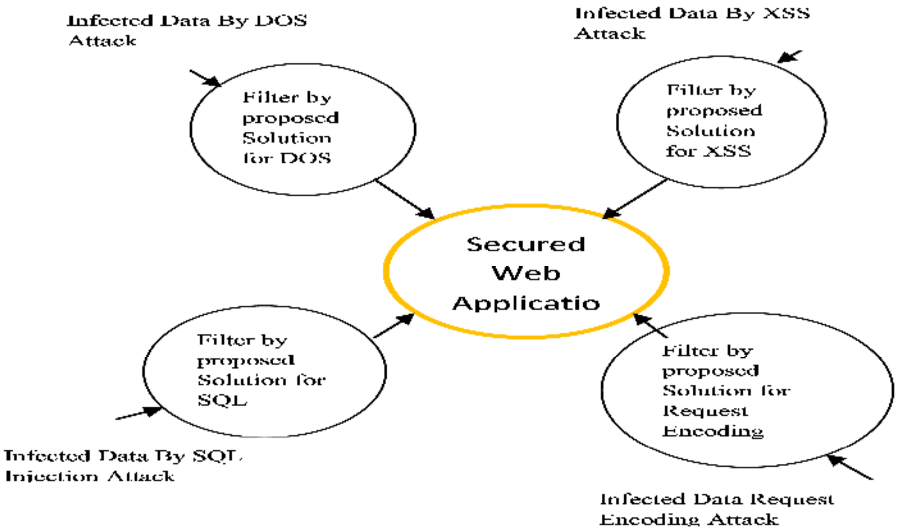


Figure 4.2 Schema diagram

4.3 Security Implementation on web


Infected Data By DOS Attack
Infected Data By XSS Attack
Filter by proposed Solution for DOS
Filter by proposed Solution for XSS
Secured Web Applicatio
Filter by proposed Solution for SQL
Filter by proposed Solution for Request Encoding
Infected Data By SQL Injection Attack
Infected Data Request Encoding Attack

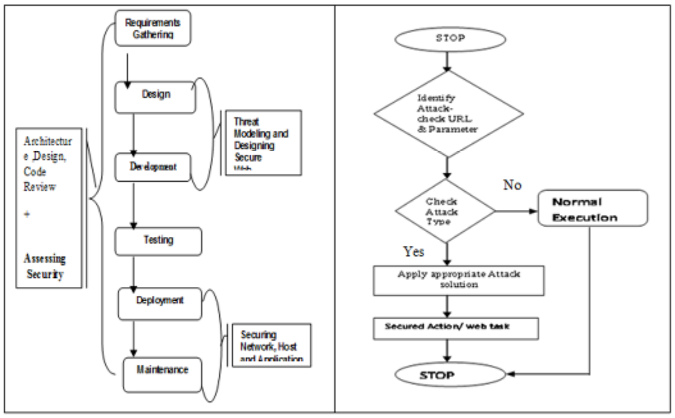

Requirements Gathering
Design
Development
Testing
Deployment
Maintenance
Architecture Design, Code Review
+
Assessing Security
Threat Modeling and Designing Secure Web
Securing Network, Host and Application
STOP
Identify Attack- check URL & Parameter
Check Attack Type
No
Normal Execution
Yes
Apply appropriate Attack solution
Secured Action/ web task
STOP

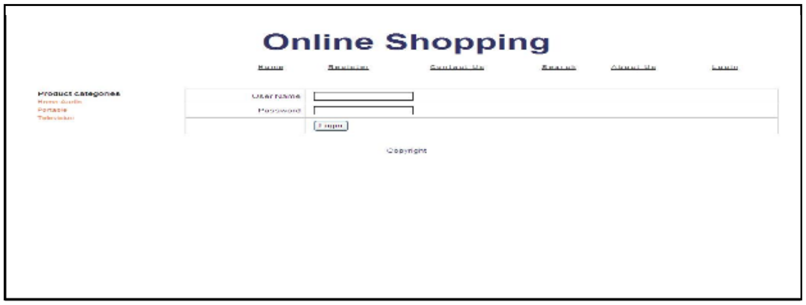
Figure 4.3 Security Implantation using E-tracking

4.4 Security maintain by Proposed E -Application

4.4.1 Terminology

E(Online)- Shopping Application	Refers to online shopping application and handling different attacks

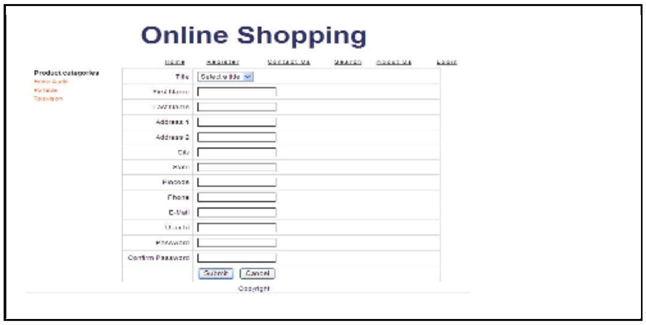
4.4.2 User interface



The screen proveides the user with different options.

- Home- This option takes the user to the main screen from any other screen.
- Register- This option allows the new user to register in to the system.
- Contact us- This option provides the contact information.
- Search – This option allows the user to search among various products
- Login – This option allows the registered users login into the system.

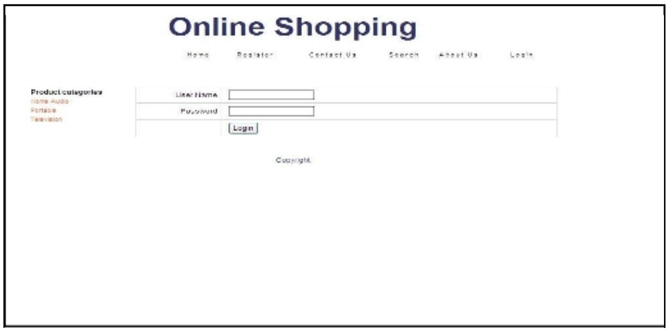
➢ Register



This screen allows the user to register into the system.

It takes different information from the user like Title, First name, Last Name, Address, City, State, PinCode, Phone, Email, UserId, Password etc. and registers the user into the system

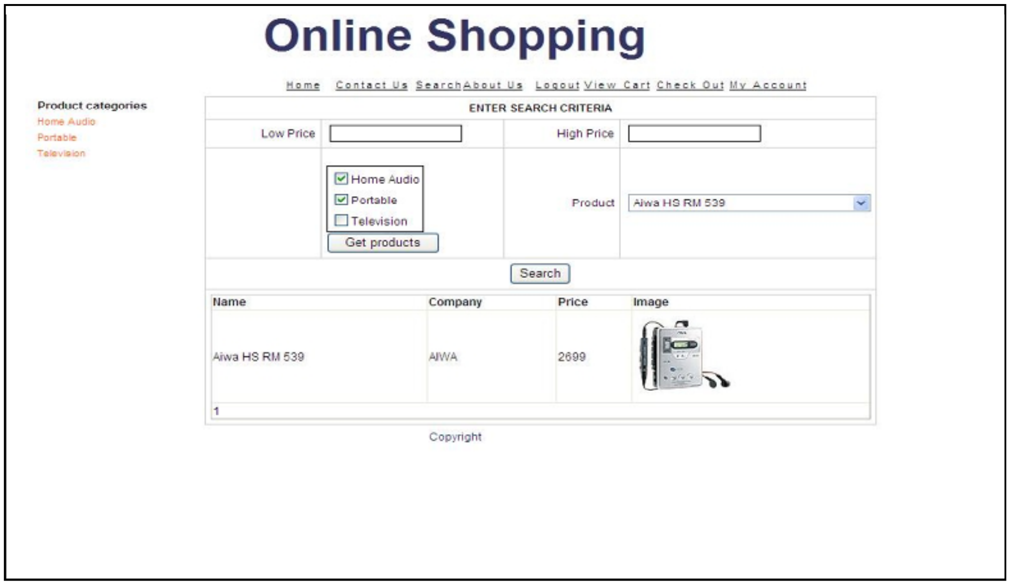
➢ Login



This screen allows the user to login into the system.

It takes User Name and Password. When click on the 'Login' button, it will validates the user name and password and if it succeed, then user is taken to the home main page.

➢ Search



Online Shopping

Home Contact Us Search About Us Logout View Cart Check Out My Account

ENTER SEARCH CRITERIA

Low Price:

High Price:

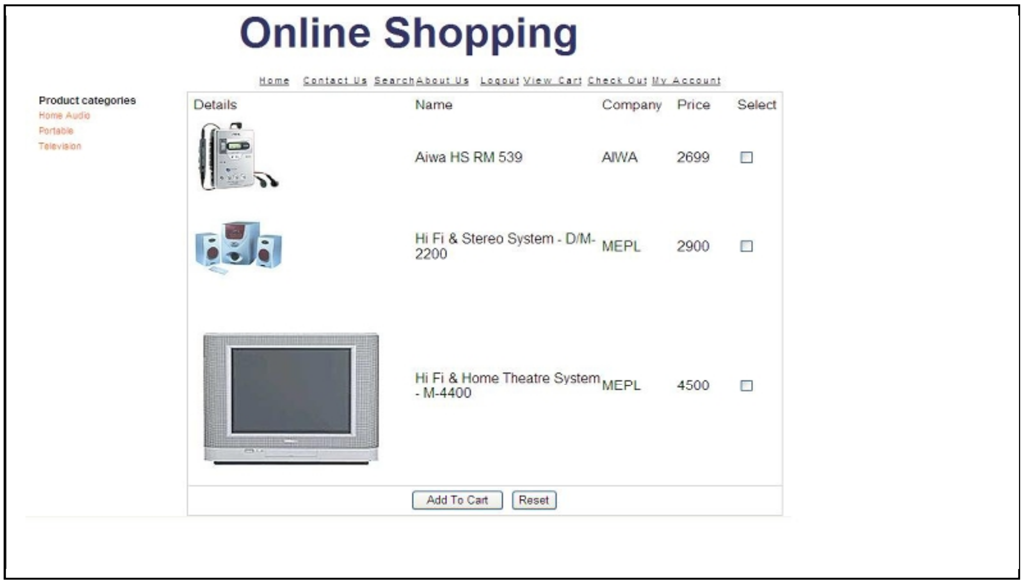
☑ Home Audio
☑ Portable
☐ Television

Product: Aiwa HS RM 539

Name	Company	Price	Image
Aiwa HS RM 539	AIWA	2699	
1			

This screen allows the user to search the products based on various selection criteria like price, Product category etc. and shows the information about the products. When user clicks on the search button, details of the searched products is displayed.

- **Product Categories**



Online Shopping

Home Contact Us Search About Us Logout View Cart Check Out My Account

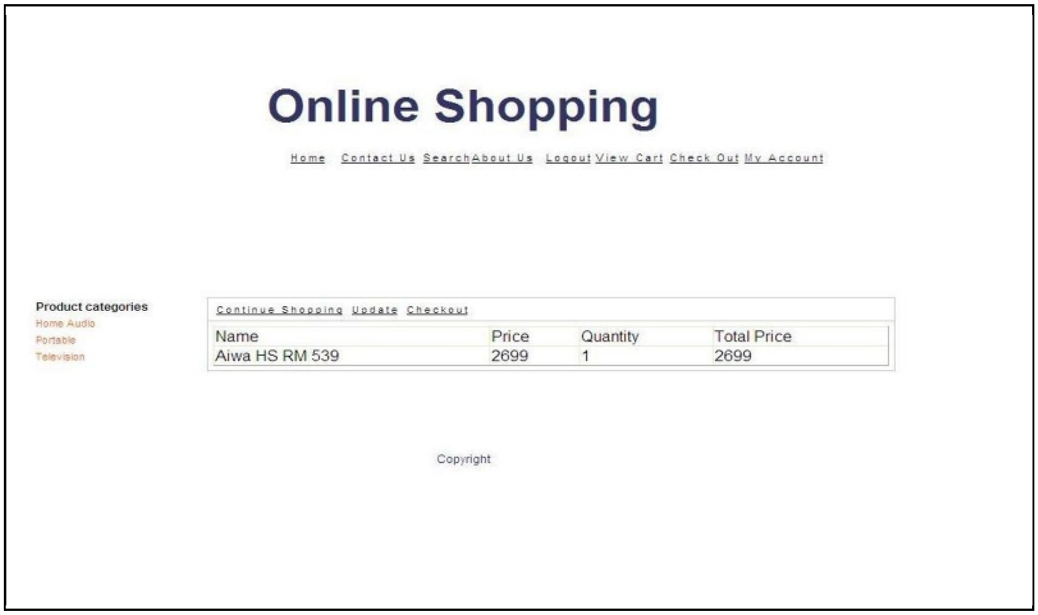
Product categories
Home Audio
Portable
Television

Details	Name	Company	Price	Select
	Aiwa HS RM 539	AIWA	2699	☐
	Hi Fi & Stereo System - D/M-2200	MEPL	2900	☐
	Hi Fi & Home Theatre System - M-4400	MEPL	4500	☐

The links on the left side of the screen takes the user to Product Categories screen. This screen all the user to select different products to add to his cart.
When user clicks o different categories, the list of products belonging to that category is displayed.

When user checks the checkbox against the product, and clicks on ‘Add to Cart’ button, Product is added into the user’s cart. User ca select multiple products to add into the cart.

- **View Cart**



Online Shopping

Name	Price	Quantity	Total Price
Aiwa HS RM 539	2699	1	2699

Copyright

This screen allows the user to view his cart details.

User is able to see the product name, Price, Quantity of the product and the total price of the product.

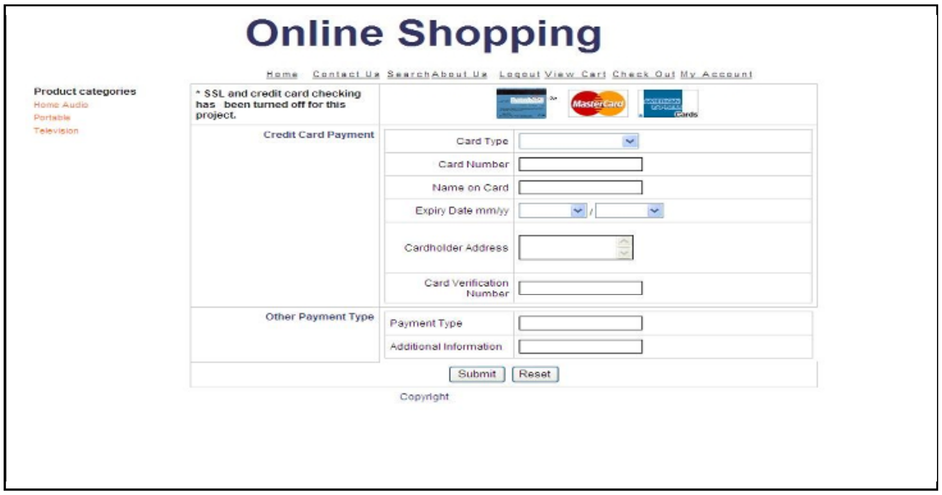
User has three options

Continue shopping – User will displayed different product details.

Update – User can update the product details like quantity.

Check Out- User can continue paying the bill for the purchased products.

- **Check Out**



Online Shopping

* SSL and credit card checking has been turned off for this project.

Credit Card Payment

Card Type

Card Number

Name on Card

Expiry Date mm/yy /

Cardholder Address

Card Verification Number

Other Payment Type

Payment Type

Additional Information

Copyright

This screen allows the user to make the payment for the purchased products.
We can apply different inputs like Credit card type, Card Number, Name on the card, Expiry date, CardHolder address, Card Verification number etc.
When user clicks on Submit button, System validates all the details and if the validation succeeded then payment for the purchased products is made successfully

V Results

For comparison we use trace net feature of asp.net for measuring the performance of secured proposed system (E-tracking) and unsecured application (same application without security) by measuring
Loading/request time & response time for modules –
Unsecuerd web application – Main- Register, Product List, View Card
Secured web application(Proposed E-tracking) - Register, Product List
All above modules are compared and we have the results in terms –
Loading time – It is the time which required to load the particular page
Response Time – It is the taken by the application to produce the results/output.
Following is the table shows the Loading time & Response Time of Unsecured web application & proposed secured web application.

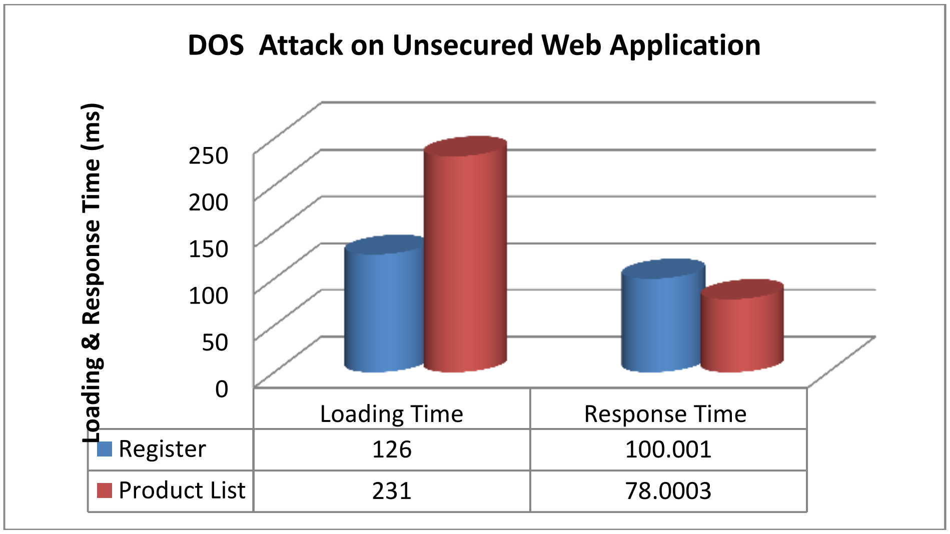
Sr. No.	**Attack Name**	**Application**	**On Module**	**Request/Loading Time**	**Response Time**
01	SQL Injection	Unsecured Web Application	Register	50 ms	46.8001 ms
			Product List	249 ms	100 ms
			View Card	210 ms	64.4002 ms
		Secured Web Application (E-Tracking)	Register	118 ms	10 ms
			Product List	240 ms	78.0003 ms
02	DOS	Unsecured Web Application	Register	69 ms	55.8001 ms
			Product List	210 ms	122 ms
		Secured Web Application (E-Tracking)	Register	126 ms	100.001ms
			Product List	231 ms	78.0003ms
03	XSS	Unsecured Web Application	Register	89 ms	99.0202 ms
			Product List	189.232	255.002 ms
		Secured Web Application (E-Tracking)	Register	1089 ms	40.0001ms
			Product List	231 ms	78.0003ms
04	Request Encoding	Unsecured Web Application	Register	156 ms	211.22 ms
			Product List	220.36 ms	352.28 ms
		Secured Web Application (E-Tracking)	Register	1089 ms	40.0001ms
			Product List	231 ms	78.0003ms

Graphical Representation of Result Analysis

By using some graphical tools we are showing the result of both unsecured web application & Secured web application (Proposed E-tracking).
We are plotting the histogram for Handling DOS,SQL Injection, XSS & Request Encoding.

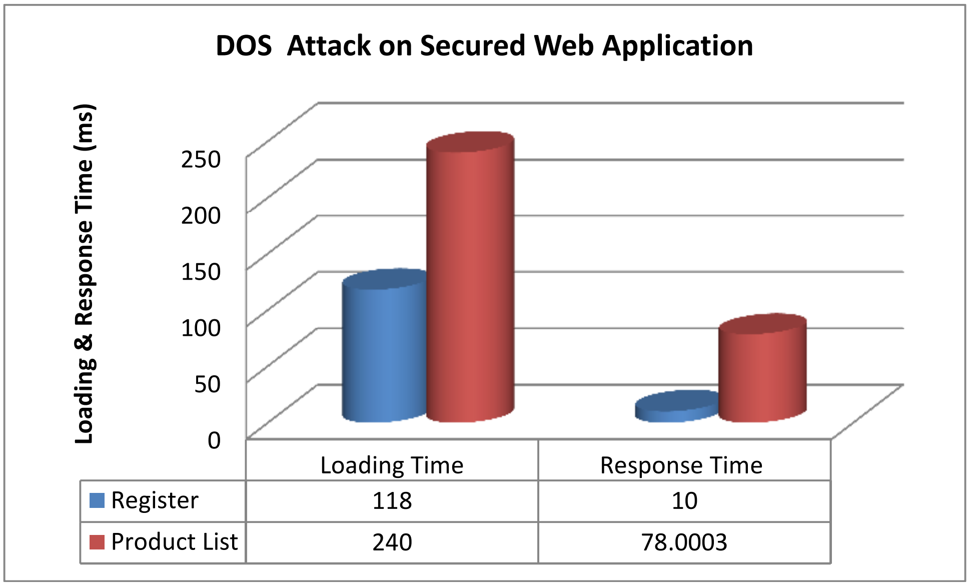
i) Measuring Performance of Secure & Unsecure Application while handling DOS attack.

- Handling DOS attack on Unsecured Web Application



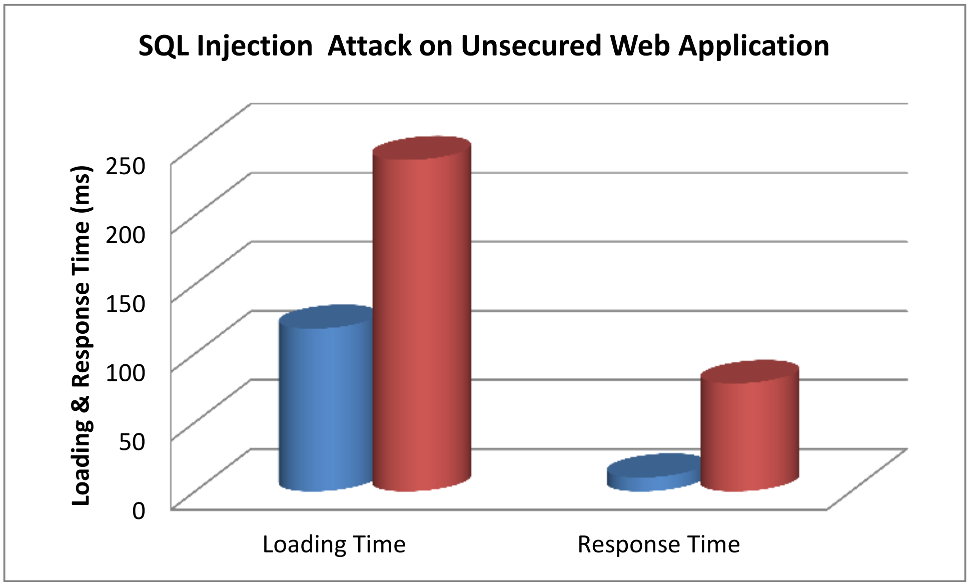
	Loading Time	Response Time
🟦 Register	126	100.001
🟥 Product List	231	78.0003

- Handling DOS attack on Secured Web Application

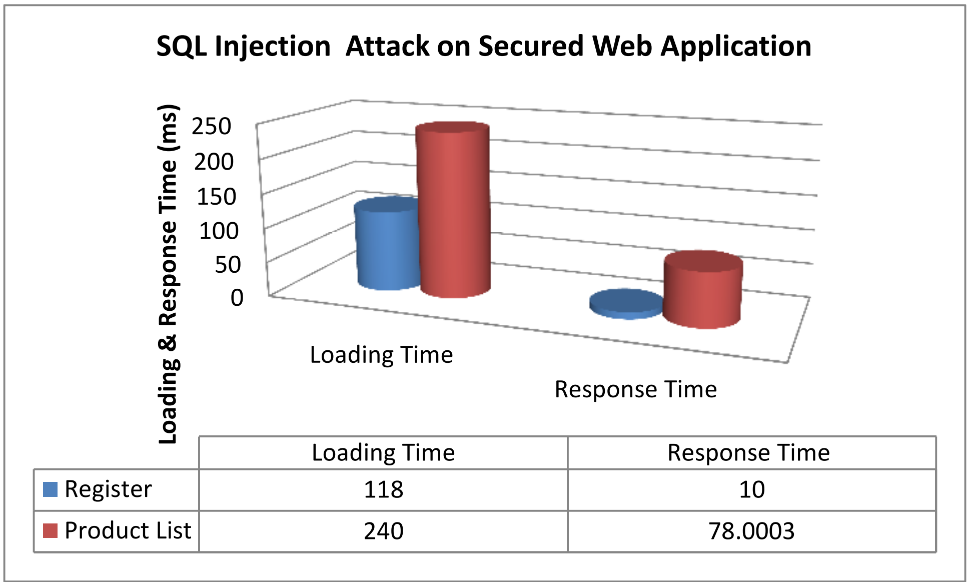


	Loading Time	Response Time
🟦 Register	118	10
🟥 Product List	240	78.0003

- Handling SQL Injection Attack on Unsecured Web Application



- Handling SQL Injection Attack on Secured Web Application



	Loading Time	Response Time
🟦 Register	118	10
🟥 Product List	240	78.0003

VI Conclusions & Future Scope

6.1 Conclusion:

The proposed solution will helpful for building rich & secured web application. We can protect the E-business world by using proposed solution. Proposed method is also gives best designing/modeling practices. The heart of the E-tracking system is protection against different web application attacks like DOS, SQL Injection, XSS and Request encoding. By using proposed mitigations for all said attacks we can make our web application secured & efficient which definitely saves our business world. Similarly we are analyzing the impaction of web attacks on secured & unsecured web application in the terms of request time, response time. With the help of results we can say we can that there is no large difference in between execution time in between both application but when we use external security for our web application like IDS etc it will take time to take for execution (Request/loading / response time) so we can apply security measures while designing and coding the web application it is very useful for protecting our web application at initial level hence there is no need to use external security measures. It will reduce the cost because security is maintained by itself.

6.2 Future Scope

This can be enhance by providing inbuilt security at primary level for large scale application which contains payment issues similarly by extending it we can secure our web application by protecting it from other web attacks. We can also provide protection against Blind SQL attack by moving in the depth of the same.

References

[1] Monika Sachdeva, Krishan Kumar Gurvinder Singh Kuldip Singh SBS College of Engg. & Technology, Guru Nanak Dev University Indian Institute of Technology Ferozepur, Punjab, India Amritsar, Punjab, India Roorkee, Uttarakhand, Indiamonika.sal(kediffmail.com gzsbawa7 1(yahoo.om kds56fec(&riitr.ernetmin) Performance Analysis of Web Service under DDoS Attacks 2009 IEEE International Advance Computing Conference (IACC 2009)Patiala, India, 6-7 March 2009

[2] Adam Kie˙zun MIT akiezun@csail.mit.edu Philip J. Guo Stanford University pg@cs.stanford.edu Karthick Jayaraman Syracuse University kjayaram@syr.edu Michael D. Ernst University of Washingtonmernst@cs.washington.edu

[3] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic, “Noxes: A clientside solution for mitigating cross-site scripting attacks,” in Proceedingsof the 12th ACM Symposium on Applied Computing, 2006.

[4] T. Gallagher, “Automated detection of cross site scripting vulnerabilities,” European Patent Application EP1420562 (pending), October 2003.

[5] Liang Guangmin Computer Engineering Department Shenzhen Polytechnic, Shenzhen 518055, China Email: gmliang@oa.szpt.net Third 2008 International Conference on Convergence and Hybrid Information Technology Modeling Unknown Web Attacks in Network Anomaly Detection.

[6] Dragan Vidakovic Gimnazija Ivanjica vidakd@ptt.yu Dejan Simic FON Belgrade dsimic@fon.bg.ac.yu A Novel Approach to Building Secure Systems

[7] Open Web Application Security Project. The ten most critical Web application security vulnerabilities. http://umn.dl.sourceforge.net/ sourceforge/owasp/OWASPTopTen2004.pdf, 2004, visit on 2005/10/05

[8] Jin-Cherng Lin and Jan-Min Chen, “An Automatic Revised Tool for Anti-malicious Injection”, in Proceedings of The Sixth IEEE International Conference on Computer and Information Technology .

[9] Igino Corona, Davide Ariu and Giorgio Giacinto This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings HMM-Web: a framework for the detection of attacks against Web applications.

[10] C. Criscione, G. Salvaneschi, F. Maggi, S. Zanero Dipartimento di Elettronica e Informazione — Politecnico di Milano 2009 European Conference on Computer Network Defense Integrated Detection of Attacks Against Browsers,Web Applications and Databases.

[11] Vipul Patel, Radhesh Mohandas and Alwyn R. Pais Information Security Research Lab, National Institute of Technology Karnataka, Surathkal, India {vip04pat, radhesh, alwyn.pais}@gmail.com ATTACKS ON WEB SERVICES AND MITIGATION SCHEMES.

[12] Forewords by Mark Curphey, Joel Scambray, and Erik Olson Improving Web Application Security Threats and Countermeasures.

[13] Encription limited The Stables White Lodge Bevere Worcester WR3 7RQ www.encription.co.uk Campbell Murray encryption limited "The need for secured web development".

[14] Jason Milletary CERT Coordination Center1 Technical Trends in Phishing Attacks

[15] SPI Dynamics 115 Perimeter Center Place Suite 270 Atlanta, GA 30346 "Blind SQL Injection" By Kevin Spett.

[16] Anatomy,Discivery,Attack,Exploitation given by Gavin Zuchinski (gav@libox.net) http:/libox.net/ November 5,2003 "The Anatomy of Cross Site Scripting "

[17] Fraser Howard, SophosLabs UK fraser.howard@sophos.com "Modern Web Attacks".

[18] StephenW. Boyd and Angelos D. Keromytis Deparment of Computer Science Columbia University fswb48,angelosg@cs.columbia.edu SQLrand: "Preventing SQL Injection Attack".

[19] Sheng-Kang Lin Inf. Syst. Lab. Inst. For Inf. Ind., Taipei, Taiwan "From Web Server to Web component security".

[20] Takesue, M. Dept. Appl. Inf., Hosei Univ.,Tokyo, Japan "An HTTP Extension for Secure Transfer of Confidential data".

[21] Tramontana ,P. Univ. di Napoli Federico II, Naples Dean, T. Tilley, S. Research Directions in Web Site Evoution II: Web Application Security.

Authors :

Ms. Vina Madhavrao Lomte [1]
Department Of Computer Engineering
, MGMCET, Kamothe,Navi Mumbai,
vinamlomte@gmail.com
9049648564

Prof. D. R. Ingle [2]
Department of Computer Engineering,
Bharrati Vidyapeeth, Kharghar,Navi Mumbai
dringleus@yahoo.com
9702777927

Prof. B. B. Meshram[3]
Department of Computer Engineering,
VJIT, Mumbai