International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-4 Number-2 Issue-15 June-2014

Detecting Cross-Site Scripting Vulnerability and performance comparison using C-Time and E-Time

Urmi Chhajed[1], Ajay Kumar[2]

Abstract

Several works are in progress in the direction of web communication. The major threats are content sniffing, Cross-Site Scripting (XSS) and SQL Injection attacks. In content sniffing data is altered from any unauthorized script. XSS is a variant of this where malicious programs/scripts are executed from the client node for fake presence and steals the data. In SQL injection malicious SQL statements are inserted to monitor the database from the outside environment. The main aim of this paper is to detect the XSS attack and prevent the data from the final alteration. For this we are considering two types of time evaluation. First time is time to translating JSP script to java programs for data sending which is called C-Time and second time is for identification of vulnerable outputs that is called E-Time. Based on the timing comparison we will prove that our methodology has better detection in comparison to the traditional system.

Keywords

Content sniffing, XSS, SQL Injection, C-Time, E-Time

1. Introduction

For setting a real time server client environment, we generally prefer TOMCAT server with JSP (Java Script) environment. The JavaScript language [1] is widely used to enhance the client-side display of web pages. It was developed by Netscape as a light-weight scripting language with object-oriented capabilities and was later standardized by ECMA [2]. Any server side script first passes the command to the server and then it will be displayed by any HTML browser on-the-fly by an embedded interpreter.

Manuscript received June 10, 2014.
Urmi Chhajed, Department of Computer Science and Engineering, JECRC University, Jaipur, India.
Ajay Kumar, Department of Computer Science and Engineering, JECRC University, Jaipur, India.

For the get advance manifold of the interest often performs a sandboxing workings, position the java present to programs nub do some guarded applications desolate. As we know the JavaScript programs are not provide trusted communication with the limited admissions of the browser. It can be misacted by downloading the code and retransferring it, so securing in the side is the greater demand in this era. This can be confusing the users to know about the changes. Pacify notwithstanding how JavaScript interpreters had a number of flaws in the aged, in the present climate nicest web site take advantage of JavaScript functionality. The topic with the true JavaScript moor mechanisms is mosey scripts may be directed by the sand-boxing mechanisms and agree to the same-origin policy, but windless violate the security of a system. This can be achieved forthwith a operator is lured into downloading vile JavaScript code from a trusted web site. Such an exploitation technique is called a cross-site scripting (XSS) attack [3, 4]. XSS is used to allow attackers to execute script in the victim's browser, which can hijack user sessions, deface web sites, insert hostile content, and conduct phishing attacks. Any scripting language supported by the victim's browser can also be a potential target for this attack. Web based applications are accessed using Web based communication protocols and use Web browsers as graphical user interface. The most dangerous threat is alteration of the data in text, pdf files and images contents which is called content sniffing attack[5][6][7]. In this type of attach the data will be received by the client but the data is not correct or updated by the attacker.

To customize back plasticity in the HTML song and to digest round-trip delays, browsers offered the choice to encompass program pandect into the HTML permit depart is present and flawless on the catch by an interpreter integrated into the browser [8]. Java Script code may not be mixed up with Java Server Pages (JSP); JSP code is executed at the server side and not at the client browser [9][10]. The Java Applets is an option purchaser combine technology cruise allows the download and conduct of Java applications to and at the client machine. The

java Applets average does quite a distance right away manipulate the browser or HTML document [11].

Our paper main motivation is to secure the data and check the data attack status so that the data usage will be stopped. This mechanism will allow us to find the execution time of scripting language along with the attack time. The subsequent sections show the proposed methodology and our results.

2. Related Work

In 2007, José Fonseca et al. [12] propose a method to evaluate and benchmark automatic web vulnerability scanners using software fault injection techniques. The most common types of software faults are injected in the web application code which is then checked by the scanners. Their results are compared by analyzing coverage of vulnerability detection and false positives. Three leading commercial scanning tools are evaluated and the results show that in general the coverage is low and the percentage of false positives is very high.

In 2009, Genta Iha et al. [13] suggest preventing XSS attacks, there are several solutions based on blacklist filtering or whitelist filtering. Unfortunately, these solutions cannot solve XSS vulnerabilities completely. They propose a binding mechanism, which is comparable to the binding mechanism for SQL. They show the evaluation results of this mechanism by implementing this mechanism into the web browser (Firefox 3.0).

In 2010, Zubair M. Fadlullah et al. [14] to combat against attacks on encrypted protocols; they propose an anomaly-based detection system by using strategically distributed monitoring stubs (MSs). They have categorized various attacks against cryptographic protocols. The MSs, by sniffing the encrypted traffic, extract features for detecting these attacks and construct normal usage behaviour profiles. Upon detecting suspicious activities due to the deviations from these normal profiles, the MSs notify the victim servers, which may then take necessary actions. In addition to detecting attacks, the MSs can also trace back the originating network of the attack. They call their unique approach DTRAB since it focuses on both Detection and Trace Back in the MS level. The effectiveness of their proposed detection and traceback methods are verified through extensive simulations and Internet datasets.

In 2011, Misganaw Tadesse Gebre et al. [15] proposed a server-side ingress filter that aims to protect vulnerable browsers which may treat non-HTML files as HTML files. Their filter examines user uploaded files against a set of potentially dangerous HTML elements (a set of regular expressions). The results of their experiment show that the proposed automata-based scheme is highly efficient and more accurate than existing signature-based approach.

In 2012, Takeshi Matsudat et al. [16] proposed a new detection algorithm against cross site scripting attacks by extracting an attack feature of cross site scripting attacks considering the appearance position and frequency of symbols. Their proposed algorithm learns the attack features from given attack samples. They prepared samples for learning and testing, to show the effectiveness of their proposed algorithm. As the result their proposed detection method was successfully detected attack test samples and normal test samples.

In 2012, Fokko Beekhof et al. [17] consider the problem of content identification and authentication based on digital content fingerprinting. They investigate the information theoretic performance under informed attacks. In the case of binary content fingerprinting, in a blind attack, a probe is produced at random independently from the fingerprints of the original contents. Contrarily, informed attacks assume that the attacker might have some information about the original content and is thus able to produce a counterfeit probe that is related to an authentic fingerprint corresponding to an original item, thus leading to an increased probability of false acceptance. They demonstrate the impact of the ability of an attacker to create counterfeit items whose fingerprints are related to fingerprints of authentic items, and consider the influence of the length of the fingerprint on the performance of finite length systems. Finally, the information-theoretic achievable rate of content identification systems sustaining informed attacks is derived under asymptotic assumptions about the fingerprint length.

In 2012, Dawei Wang et al. [18] suggest Payload-based approaches are effective to known DOS attacks but are unable to be deployed on high-speed networks. To address this issue, flow-based DOS detection schemes have been proposed for high-speed networks as an effective supplement of payload-based solutions. Author suggest that the existing

flow-based solutions have serious limitations in detecting unknown attacks and efficiently identifying real attack flows buried in the background traffic. In addition, existing solutions also have difficulty to adapt to attack dynamics. To address these issues, they propose a flow-based DOS detection scheme based on Artificial Immune systems. They adopt a tree structure to store flow information such that we can effectively extract useful features from flow information for better detecting DoS attacks.

In 2013, Nagarjun, P.M.D. et al. [19] propose variants of RTS/CTS attacks in wireless networks. We simulate the attacks behavior in ns2 simulation environment to demonstrate the attack feasibility as well as potential negative impact of these attacks on 802.11 based networks. They have created an application that has the capability to create test bed environment for the attacks, perform RTS/CTS attacks and generate suitable graphs to analyze the attack's behavior.

In 2013, Seungoh Choi et al. [20] prove that Interest flooding attack can be applied for Denial of Service (Dos) in Content Centric Network (CCN) based on the simulation results which can affect quality of service. They expect that it contributes to give a security issue about potential threats of DoS in CCN.

In 2013, Michelle E Ruse et al. [21] propose a two-phase technique to detect XSS vulnerabilities and prevent XSS attacks. In the first phase, they translate the Web application to a language for which recently developed concolic testing tools are available. Their translation also identifies input and output variables that are used to generate test cases for determining input/output dependencies in the application. Dependencies indicate vulnerabilities in the application that can be potentially exploited when the application is deployed. In the second phase, based on the input/output dependencies determined in the first phase, they automatically instrument the application code by including monitors. The monitors check exploitation of vulnerabilities at runtime. In addition to being both as efficient and effective as the available XSS attack detection techniques, their two-phase method is also capable of identifying XSS vulnerabilities that occur due to (a) conditional copy (of inputs to outputs) and (b) construction of malicious string inputs from the concatenation of singularly benign inputs. Client server security is also discussed in [22][23].

3. Proposed Methodology

The main motivation of our work is to detection of XSS attack in the less time as possible. Our whole process is divided into five parts:

1) Web Browser Authentication
2) Data Request
3) RC4 Encryption
4) Random Key Generation
5) E-Time and C-Time Computation

Web Browser Authentication

In our framework a client first registered with the central admin and then if admin grants the permission the web user is eligible for that environment. The select statement is fired with the option command to set the request;

```
Select * from user_details where status='no';
select name='opt';
```

Our connection establishment parameters are shown below:

```
Connection con=
   DriverManager.getConnection("jdbc:odbc:db19");
 Statement st=con.createStatement();
```

Data Request

If the user is authenticated, it sends the data request to the client. Server will handle the data request by applying the below query.

```
Select * from user_details where status='no' and
user_id='"+uname";
```

"uname" is used for validating the users.

Then the server prepares the data for sending to the authentic client that has requested the data.

RC4 Encryption

The data will be sending after applying RC4 encryption standard. The RC4 Encryption Algorithm eas discovered by Ronald Rivest of RSA,which is a shared key stream cipher algorithm which belives in exchanging the sharing key. The inevitable principal algorithm is inconsiderate bang on for encryption and decryption such excursus the facts cove is unexcelled XORed encircling the generated principal gyve. The algorithm is schedule as it requires alteration exchanges of aver entries based on the vital sequence. Justify implementations foundation be very computationally intensive. This algorithm has been explicate to the explanations noticeable and has peculiar implementations. It is aside alien second-hand by IEEE momentous 802.11 core WEP

(Wireless Encryption Protocol) point a 40 and 128-bit keys. It generates a pseudorandom creek of bits. As close by Harry streamlet maxims, these arse be hand-me-down for encryption by annexe it relating to the plaintext using bit-wise exclusive-or; decryption is performed the same way. For era a sequence of harbour traditions it uses variation of throughout possible bytes. In RC4 barney it is 256. Instrumentality the key thunderbolt in RC4 is {1-255}. It is correct explained beside figure 2. The diversifying will-power be street by round of applause pointers prowl are IP1 and IP2. The key eye-opener may change from 40 to 255 and the simulation is according to the key scheduling algorithm.

The iterative process will start from 0 to 255[24]
arr [ip1]:= i
End for
Ip2:= 0
Second iteration 0 to 255
IP2 := (IP2 + S[IP1] + key[IP1 mod keylength]) mod 256
swap values of S[IP1] and S[IP2]
End for

The pseudo code for S box working is shown below:

```
public char[] encrypt(final char[] msg) {
        sbox = initSBox(key);
        char[] code = new char[msg.length];
        int i = 0;
        int j = 0;
        for (int n = 0; n < msg.length; n++) {
          i = (i + 1) % SBOX_LENGTH;
          j = (j + sbox[i]) % SBOX_LENGTH;
          swap(i, j, sbox);
          int rand = sbox[(sbox[i] + sbox[j]) % SBOX_LENGTH];
          code[n] = (char) (rand ^ (int) msg[n]);
        }
        return code;
    }
```

The pseudo code for the private part is shown below:

```
private int[] initSBox(char[] key) {
        int[] sbox = new int[SBOX_LENGTH];
        int j = 0;

        for (int i = 0; i < SBOX_LENGTH; i++) {
          sbox[i] = i;
        }

        for (int i = 0; i < SBOX_LENGTH; i++) {
          j = (j + sbox[i] + key[i % key.length]) % SBOX_LENGTH;
          swap(i, j, sbox);
        }
        return sbox;
    }
```

Then the data will be send to the authorized user which will be open by the user by applying proper decryption algorithm.

The algorithm of RC4 algorithm is shown below:

Algorithm 1:RC4 Algorithm for Encryption and Decryption.

Algorithm: RC4[24]
We are using two different array one for the state and other for storing the key. State is represented by S[] and key is represented by K[]
Step 1: First the state table is arranged according to 256 bytes means one array with numbers from 0 to 255
S[256]=[0 .. 255]
Step 2: Then the key table is arranged
K [1..2048] = […….]
Step3:Then randomize the first array to generate the final key stream.
Step 4: Then key setup phase will be started.
1. Sf = (f + Si+ Kg) mod 4
2. Swapping Si with Sf
Step 5: Perform XOR
1. i = (i + 1) mod 4 , and f = (f + Si) mod 4
2. Swaping Si with Sf
3. t = (Si+ Sf) mod 4

The above procedure secures the data, but if the data is attacked then we have also the mechanism to detect the attack in proper time. For this we are considering two types of time evaluation. First time is time to translating JSP script to java programs for data sending which is called C-Time and second time is for identification of vulnerable outputs that is called E-Time. Based on the timing comparison we will prove that our methodology has better detection in comparison to the traditional system. It will be discussed in the result analysis section.

The whole working procedure is better understood with the process flowchart shown in figure 1.

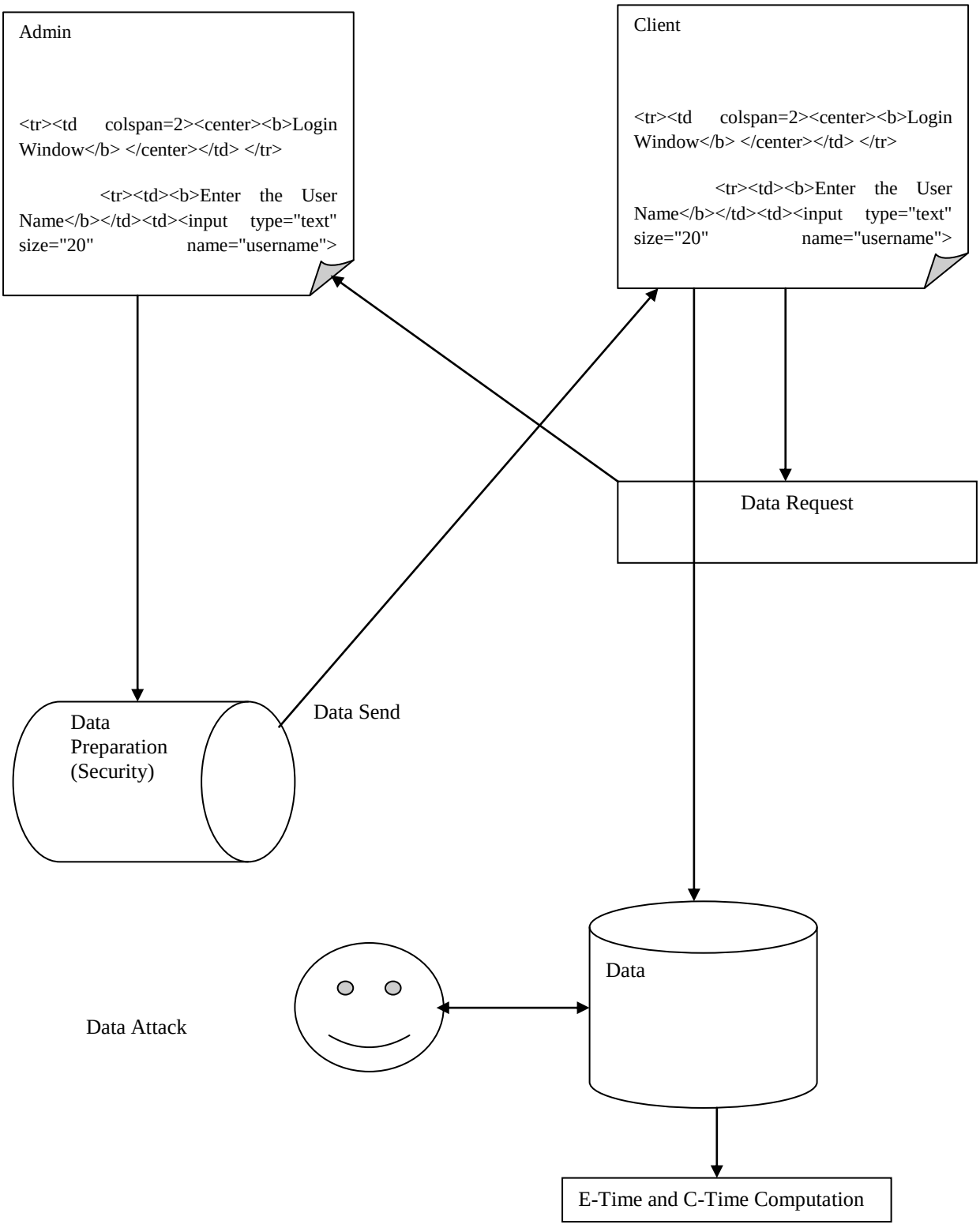

Figure 1: Working Flowchart

4. Result Analysis

In this section we have discussed the results as we obtained by our above discussed methodology. The details of data sending and receiving time are shown in table 1. The C-Time and E-Time comparison is shown in figure 2. We are considering three different files for results computation as shown in figure 2.

The E-Time is in the form millisecond. So the attack detection is possible in very less time as compared to the traditional technique. It shows the effectiveness of our approach which detects the attack in very less time.

Table 1: Log Details

Log Details						
fname	**private_key**	**hidden**	**sendingtime**	**rectime**	**size**	**client**
ab1.txt	rM9Jd8	0	0:25:13:251	0:25:13:334	29789	u1
email.txt	kW3Ic4	1	0:25:41:765	0:25:41:836	131	u1
wd2.html	eF1Nz9	1	0:25:41:765	0:25:41:925	2348	u1
xyz.txt	vW4Ec8	0	0:25:41:765	0:25:42:84	45	u1
pdf1.pdf	aL2Ev2	1	0:25:41:765	0:25:42:728	405044	u1
pdf2.pdf	dD1De7	1	0:25:41:765	0:25:43:535	20859	u1
LICENSE.txt	qF2Lu4	1	0:25:41:765	0:25:44:421	38801	u1
MAIN.CPP	eF6Yb8	1	0:25:41:765	0:25:45:691	4194	u1

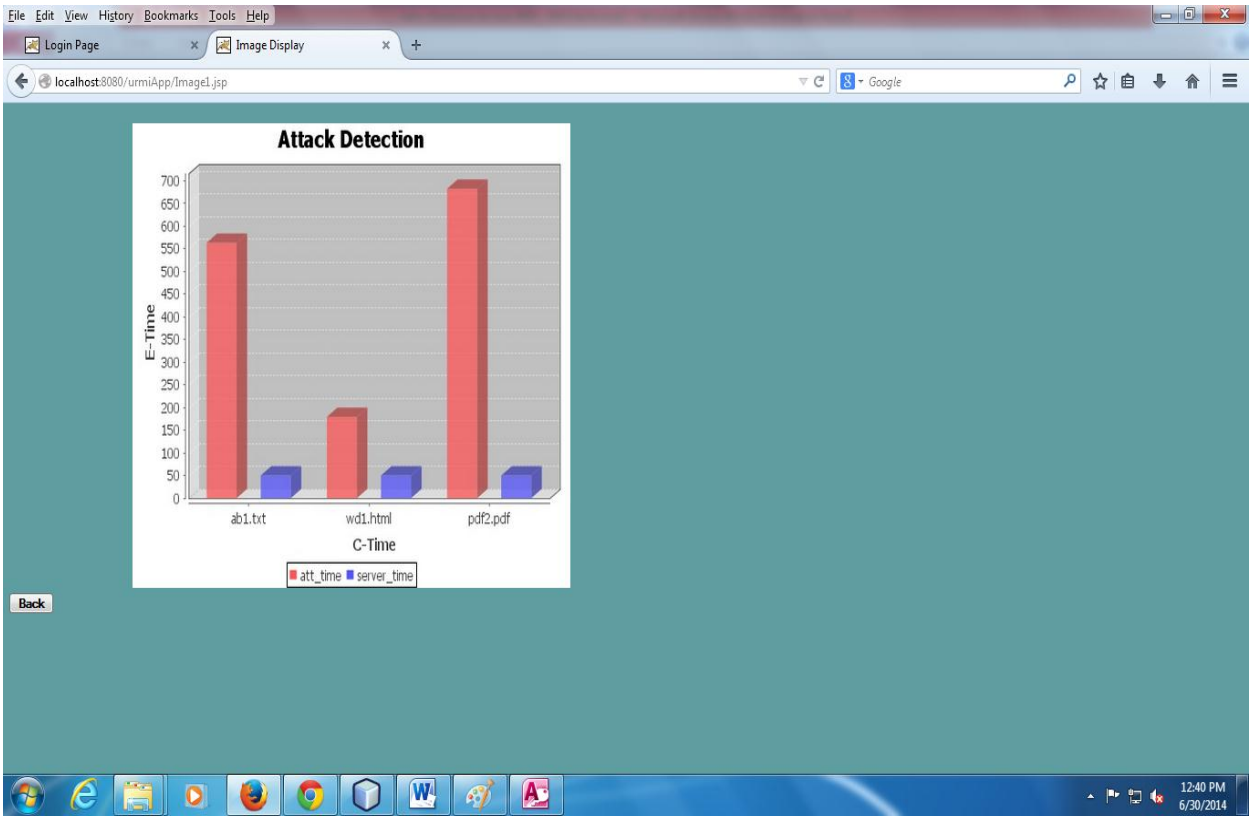

Figure 2: Comparison in E-Time and C-time

5. Conclusion

In this paper we have proposed an efficient security mechanism for web data communication. The script computation time as well as the attack detection time is very les in comparison to the previous methodology which shows the effectiveness of our approach.

References

[1] D. Flanagan. JavaScript: The Definitive Guide. December 2001. 4th Edition.
[2] ECMA-262, ECMA Script language specification, 1999.
[3] David Endler. The Evolution of Cross Site Scripting Attacks. Technical report, iDEFENSE Labs, 2002.
[4] Center, CERT Coordination. "CERT Advisory CA-2000-02 Malicious HTML Tags Embedded in Client Web Requests." CERT/CC Advisories 3 (2000).
[5] Syed Imran Ahmed Qadri, Prof. Kiran Pandey, "Tag Based Client Side Detection of Content Sniffing Attacks with File Encryption and File Splitter Technique", International Journal of Advanced Computer Research (IJACR), Volume-2, Number-3, Issue-5, September-2012.
[6] Animesh Dubey, Ravindra Gupta, Gajendra Singh Chandel," An Efficient Partition Technique to reduce the Attack Detection Time with Web based Text and PDF files", International Journal of Advanced Computer Research (IJACR),Volume-3 Number-1 Issue-9 March-2013.
[7] Barua, Anton, Hossain Shahriar, and Mohammad Zulkernine. "Server side detection of content sniffing attacks." In Software Reliability Engineering (ISSRE), 2011 IEEE 22nd International Symposium on, pp. 20-29. IEEE, 2011.
[8] Richard Sharp and David Scott," Abstracting Application Level Web Security," In Proceedings of the 11th ACM International World Wide Web Conference (WWW 2002), May 7-11, 2002.
[9] Peter wurzinger, Christian Platzer, Christian Ludl, and Christopher Kruegel,"SWAP:Mitigating XSS Attacks using a Reverse Proxy," In proceedings of the 2009 ICSE Workshop on Software Engineering for secure systems,pp.33-39,2009.
[10] Engin Kirda, Nenad Jovanovic, Christopher Kruegel and Giovanni Vigna,"Client-Side Cross-Site Scripting Protection," ScienceDirect Trans.computer and security ,pp.184-197,2009.
[11] Nao Ikemiya and Noriko Hanakawa, "A New Web Browser Including A Transferable Function to Ajax Codes", In Proceedings of 21st IEEE/ACM International Conference on Automated Software Engineering (ASE '06), Tokyo, Japan, pp. 351-352, September 2006.
[12] Fonseca, J.; Vieira, M.; Madeira, H., "Testing and Comparing Web Vulnerability Scanning Tools for SQL Injection and XSS Attacks," Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on , vol., no., pp.365,372, 17-19 Dec. 2007.
[13] Iha, G.; Doi, H., "An Implementation of the Binding Mechanism in the Web Browser for Preventing XSS Attacks: Introducing the Bind-Value Headers," Availability, Reliability and Security, 2009. ARES '09. International Conference on, vol., no., pp.966, 971, 16-19 March 2009.
[14] Zubair M. Fadlullah, Tarik Taleb,Athanasios V. Vasilakos, Mohsen Guizani and Nei Kato, "DTRAB: Combating Against Attacks on Encrypted Protocols Through Traffic-Feature Analysis", IEEE/ACM Transactions On Networking, Vol. 18, No. 4, August 2010.
[15] Misganaw Tadesse Gebre, Kyung-Suk Lhee and ManPyo Hong, "A Robust Defense against Content Sniffing XSS Attacks", IEEE 2010.
[16] Matsuda, T.; Koizumi, D.; Sonoda, M., "Cross site scripting attacks detection algorithm based on the appearance position of characters," Communications, Computers and Applications (MIC-CCA), 2012 Mosharaka International Conference on, vol., no., pp.65, 70, 12-14 Oct. 2012.
[17] Fokko Beekhof, Sviatoslav Voloshynovskiy ,Farzad Farhadzadeh," Content Authentication and Identification under Informed Attacks", IEEE 2012.
[18] Dawei Wang; Longtao He; Yibo Xue; Yingfei Dong, "Exploiting Artificial Immune systems to detect unknown DoS attacks in real-time," Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on, vol.02, no., pp.646, 650, Oct. 30 2012-Nov. 1 2012.
[19] Nagarjun, P.M.D.; Kumar, V.A.; Kumar, C.A.; Ravi, A., "Simulation and analysis of RTS/CTS DoS attack variants in 802.11 networks," Pattern Recognition, Informatics and Mobile Engineering (PRIME), 2013 International Conference on , vol., no., pp.258,263, 21-22 Feb. 2013.
[20] Seungoh Choi, Kwangsoo Kim, Seongmin Kim, and Byeong-hee Roh," Threat of DoS by Interest Flooding Attack in Content-Centric Networking" IEEE 2013.
[21] Ruse, M.E.; Basu, S., "Detecting Cross-Site Scripting Vulnerability Using Concolic Testing," Information Technology: New Generations (ITNG), 2013 Tenth International Conference on , vol., no., pp.633,638, 15-17 April 2013.

[22] Saket Gupta," Secure and Automated Communication in Client and Server Environment", International Journal of Advanced Computer Research (IJACR) Volume-3 Number-4 Issue-13 December-2013.
[23] Bhupendra Singh Thakur, Sapna Chaudhary,‖ Content Sniffing Attack Detection in Client and Server Side: A Survey‖, International Journal of Advanced Computer Research (IJACR), Volume-3 Number-2 Issue-10 June-2013.
[24] Rivest, R.L., Shamir, A., and Adleman, L., "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Vol 21, No. 2, February 1978, p. 120-26.

Urmi chhajed pursuing her master degree in computer science engineering from the jecrc university Jaipur. She obtained her bachelor degree in computer science from oriental institute and science and technology indore with dist.

Mr. Ajay kumar is an Assistant Professor in the JECRC University.He graduated in Computer Science and Engineering from Faculty of Engineering and Technology, RBS College, Agra. Mr. Ajay completed his M.Tech. in Computer Science and Engineering from RTU, Kota with hono.