Research Article **ISSN: 2395 – 5597**

International Journal of Engineering and Robot Technology

Journal home page: www.ijerobot.com

AN ELECTRONIC VOTING SYSTEM TO SEGUARD ELECTORAL INTEGRITY

Edward Danso Ansong[1]*, Dominic Damoah[1], Richard Asante[2]

[1*]Department of Computer Science and Engineering, Kwame Nkrumah University of Science and Technology, Ghana, West Africa.

[2]Department of Computer Science and Engineering, Valley View University, Techiman, Ghana, West Africa.

ABSTRACT

Electronic voting has become a major interest to many countries worldwide. The most sensitive part of electronic voting is its security issues that have become a national issue in the world in developed and developing countries. The worst case plan is catastrophic when election results are without integrity. Electronic voting systems will provide accurate counting, results, results of transmission time and electoral processes are secured when it is implemented properly. This paper describes how to protect an electronic voting system against vulnerabilities like Structured Query Language (SQL) Injection, Session Hijacking, and Cross-Site Scripting. An electronic voting system was developed to test and implement how an electronic voting system can be more secured.

KEYWORDS

Session hijacking, SQL injection, Cross-Site Scripting, Your Right, Encryption algorithm, Counter measures and anonymous.

Author for Correspondence:

Edward Danso Ansong,

Department of Computer Science and Engineering, Kwame Nkrumah University of Science and Technology, Ghana, West Africa.

Email: edkan20002002@yahoo.com

INTRODUCTION[1]

Our electoral processes and democracy have been in jeopardy since the introduction of e-voting systems[2]. The reason is that most these e-voting systems are susceptible to security vulnerabilities like SQL injection, Session hijacking, and Cross-Site Scripting. Hackers are able to intercept results and manipulate them to suite their political agenda when these issues are not addressed. Anonymity enforces all voters to cast vote in their own free will and how a voter voted should not be known by any one[3]. This paper attempts to describe the aforementioned security vulnerabilities in e-voting systems and their countermeasures by designing e-voting system called Your Right. Your Right is an electronic voting system (e-voting) that would use secure socket layer (SSL) which is a security technology for establishing an encrypted link between a server and a client typically webserver and a browser. The communication between voters and voting centres would be secured using SSL.

Your Right voting system would satisfy the following requirements:

- Individuals who are not registered would not be able to vote.
- Every voter would vote once to ensure transparency.
- During voting, votes have to be kept confidential.
- How voters cast their votes must not be proven in anyway.
- The total votes cast must correspond to the results of the votes counted.
- Additional ballots cannot be cast once the election is over.
- Under no circumstances can a voter prove how they voted.
- The system would address e-voting vulnerabilities like Cross-Site Scripting (XSS), SQL Injection, and Session Hijacking.
- An encryption algorithm would be designed to protect votes and voters identity.

RELATED WORK

Many systems exist in the area of secure electronic voting to make our electoral processes transparent. An anonymous electronic voting scheme which can be used in real-world elections was presented by Chang and Lee in the year 2006. Chang and Lee's e-voting scheme were susceptible to security attacks like SQL injection, Cross Site Scripting (XSS), Session Hijacking, and Session Sniffing. The aforementioned security vulnerabilities were not checked to mitigate e-voting vulnerabilities and this may compromise their e-voting scheme.

In[3] the authors made emphasis on communication channel which is secured under "Secure Socket Layer (SSL)" infrastructure. They proposed SSL application to secure the communication between voting centres and voters.

In[4] the authors used deniable encryption to prevent rigging in e-voting and e-auction. They used this deniable encryption to prevent coerciveness and eavesdropping of communication channel to allow candidates and voters to cast their votes privately.

The integration of third-party services into web application has increasingly become prevalent. New security challenges have emerged since the integration and to make an application work together with its internal states of web clients and component services across internet complicate the application[5].

Hardware security, software security, and environment are the main vulnerable areas in web applications. Any devices that are used for running e-voting web application like servers and network devices relate to hardware security. Network security is protecting network devices by using properly configured firewall device that would only allow needed ports to access e-voting web applications. Networks using a network DMZ must be separated from servers such as database servers and web servers to reduce likely intrusion from compromised computers that are found behind firewalls on other networks. A separate network that is in between an external network and a protected network, in order to provide an additional layer of security is called demilitarized zone or DMZ[6].

Web server software (IIS, Apache), operating system, and database software that are used in running e-voting applications are part of software security. The security of the operating system must well be configured to harden the operating system. When patches and security fixes are released the software should be updated frequently. The web application should indeed be hardened against cookie poisoning, hidden-filed manipulation, parameter tampering, cross-site scripting, SQL injection, and buffer overflow which are common attacks[6].

The environment security is protecting e-voting applications' hardware against human resources. Physical access to network and server devices that run the e-voting system must be well protected physically and their credentials should be highly complex[6].

Vulnerabilities exist in servers and can be exploited by hackers with criminal intentions who are determined to cause destructions or acquire information illegally. Web servers respond to HTTP requests when delivering web pages. Software is more susceptible to coding errors (bugs) and security holes when it is complexly designed. Security holes are loop holes or weaknesses in systems that give way to hackers to do evil things[7].

SYSTEM DESIGN

The main goal of the system design is to come up with a perfect solution which would satisfy the functional requirements for the system. System modelling and database design would be shown at the system requirements analysis phase which would show the physical architecture of the system.

Requirements of E-voting System

The design of any voting system should depend on a number of competing criteria. The requirements would make elections free and fair, credible and confidential[8].

From the functional and non-functional point of view, e-voting system must follow the famous CIA security triangle consists of three main sides which are Confidentiality, Integrity, and Availability.

Confidentiality

Ensuring that unauthorised users do not get access to sensitive data. Cryptographic techniques like Kerberos and SSL channels are importantly used to achieve confidentiality. Legitimate users are the only users granted access with the help of access control[9]. With these implementations, illegitimate users and attackers cannot cast votes during elections.

Integrity

Integrity when data cannot be tempered with or altered giving the assurance of originality. For integrity to be effective, or to be protected, cryptographic techniques like SSL, TSL, and MAC must be implemented. Integrity is all about trustworthiness of both data and functions. Safeguarding integrity means that data and software would not be altered in an unauthorised way. Data and software integrity are the two categories of integrity. The preservation of ballot information and audit records is called data integrity. Software integrity ensures that only genuine and authentic pieces of software runs on the system components of electronic voting system[9]. Accuracy must be ensured which indicates that only votes cast are to be calculated. After and during elections, system behaviour can be traced to ensure auditability[10].

Availability

Availability is when a system is fully or partially functional when it is needed. Attacks, system malfunctions, resources overloading are some problems that could cause a system to be unavailable[10]. A system's availability can be conserved when it is capable of failure resiliency and Denial of Service (DoS) attack counter measures[9].

Entity Relationship Diagram

The base of a management information system is database system. Relational databases have become prevalent and are used widely in various areas[11]. According to[11] "Entity-Relationship Diagram (ERD) is a common technique for data structures and database systems design".

SYSTEM TESTING AND HACKING

Testing for SQL Injection

SQL Injection has become prevalent and one of the most effective ways of getting sensitive information from a database. When a system or web application becomes susceptible to SQL Injection hackers exploit that vulnerability and steal sensitive information from a database[12]. Form February 2015 Cyber Attacks Statistics, it is clear that, SQL injection is the most serious and mostly used type of cyber-attack performed these days when compared to other attacks[13].

With the SQL Injection attack being the most serious and mostly used by attackers the need has arisen to protect web applications to make them safe and more secured to ensure transparency in systems especially an electronic voting system like Your Right.

On the off chance that the data submitted from the program to the Your Right voting framework is embedded into a database query without being checked legitimately, then there may be a possibility of event of SQL Injection. HTML structure that gets and passes the data presented by the client on the Hypertext Preprocessor (PHP) script running on Apache web server is the best illustration of SQL Injection. The data passed is the client's authentication credentials.

Username: Blah ' or 1 = 1 --
Password: Springfield
The query executed is:
SELECT Count (*) FROM Users WHERE User Name = 'Blah' or 1 = 1-- AND
Password = 'Springfield‘;

Attackers are able to use combination of these characters to detect SQL Injection and exploit them. Below image shows most of the characters used by attackers for exploiting SQL Injection vulnerabilities in web applications[14].

How Your Right voting system was defended against SQL Injection

Function to sanitize values received from the forms. Prevents SQL injection

```
function clean ($str) {
$str = @trim ($str);
if (get_magic_quotes_gpc()) {
$str = stripslashes ($str);
}
returnmysql_real_escape_string ($str);
}
```

Session hijacking

A series of attacks related to web applications are referred to as session hijacking. It is a situation where by an attacker steals a session credential and gains access to a web application by impersonating the user of the web application or system[15].

When a user logs onto the server, an attacker steals or captures the Session ID and then gains an unauthorised access to the server by using the valid token session called “Session ID”.

Attackers use sniffers to capture Session ID when the HTTP traffic sent are not encrypted and the HTTP traffic could contain the user's authentication credentials. The attacker sniffs a legitimate Session ID=ACF3035F216AAEFC and logs into the web application as a legitimate user. When this attack is successful, the attacker can do anything that the legitimate user can do on the system or web application[16].

To protect against session hijacking, defence-in-depth mechanisms have to be implemented and sessions have to be destroyed when they expire. A perfect example of defence-in-depth is a firewall configuration[16].

The following lines of code in red colour Protect against Session hijacking in the Your Right voting system[23]

```
functionpage_protect() {
session_start();
global $db;
functionpage_protect() {
session_start();
global $db;
```

Secure against Session Hijacking by checking user agent[23]

```
If (isset($_SESSION['HTTP_USER_AGENT']))
{
If ($_SESSION['HTTP_USER_AGENT'] != sha1($_SERVER['HTTP_USER_AGENT']))
  {
Logout ();
exit;
  }
}
```

Before we allow sessions, we need to check authentication key - ckey and ctime stored in database.

If session not set, check for cookies set by Remember me[23]

```
If (!isset($_SESSION['user_id']) && !isset($_SESSION['user_name']) )
{
        If(isset($_COOKIE['user_id']) &&isset($_COOKIE['user_key'])){
```

We double check cookie expiry time against stored in database[23]

```
        $cookie_user_id= filter($_COOKIE['user_id']);
        $rs_ctime = mysql_query("select `ckey`,`ctime` from `users` where `id` ='$cookie_user_id'") or die(mysql_error());
        list($ckey,$ctime) = mysql_fetch_row($rs_ctime);
        // coookie expiry
        if( (time() - $ctime) > 60*60*24*COOKIE_TIME_OUT) {
                logout();
                }
```

Security check with untrusted cookies – do not trust value stored in cookie. We also do authentication check of the `ckey` stored in cookie matches that stored in database during login[23].

```
        If( !empty($ckey) &&is_numeric($_COOKIE['user_id']) &&isUserID($_COOKIE['user_name']) && $_COOKIE['user_key'] == sha1($ckey)  ) {
                session_regenerate_id(); //against session fixation attacks.
        $_SESSION['user_id'] = $_COOKIE['user_id'];
  $_SESSION['user_name'] = $_COOKIE['user_name'];
```

Query user level from database instead of storing in cookies[23]

```
list($user_level) = mysql_fetch_row(mysql_query("select user_level from users where id='$_SESSION[user_id]'"));
         $_SESSION['user_level'] = $user_level;

$_SESSION['HTTP_USER_AGENT'] = md5($_SERVER['HTTP_USER_AGENT']);
         } else {
        logout();
          }
  } else {
        header("Location: login.php");
        exit();
        }
}
}
```

Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) attacks is the third most severe attacks in the OWASP Top 10 2013 list and XSS affects 53% of web applications and that makes them vulnerable[17].

Cross-Site Scripting is when a malicious scripts are inject into a web applications. An attacker would send a malicious code to the end user or the victim and when the end user executes the malicious code, the attacker can access any cookies, session tokens, and other confidential information and use it against the site or the web application[18].

To protect against Cross-Site Scripting (XSS) user input has be handled:

- Uniform Resource Identifier (URI) must be sanitized APIs private sensitive activities has be isolated HTML character encoding has to be implemented[19].
- APIs private sensitive activities has be isolated HTML character encoding has to be implemented[19].
- HTML character encoding has to be implemented[19].
- .With the html character encoding the HTML tags would be converted when the attacker injects those malicious codes in to the HTML forms[20].

Encryption Algorithm

Cryptography is means protection of confidential data. Confidential data could be personal data, corporate data, chat sessions, email messages, web transactions and many others. Many people do have secrets to share with their loved ones, colleagues, co-workers, partners and other people. In terms of sharing these secrets which are mostly confidential data, it has to be done digital and sending data digitally brings about some security concerns. So the data or information being transferred has been protected during transfer. In order to prevent other people from reading or stealing the data or information, encryption has to be implemented. Encryption converts a plaintext into an unreadable text (cipher text) to maintain the security of the data being transferred. Ciphers are algorithms that used to encrypt and decrypt data to maintain its security and integrity. Examples of encryption algorithms are AES (Advanced Encryption Standard), DES (Data Encryption Standard), RC4, RC5, RC6, and blowfish algorithms[21].

With the electronic voting system (Your Right), blowfish encryption algorithm was used and a salt was appended to it in order to make more secure. Bruce Schneier designed the blowfish algorithm and published it in 1994. The blowfish is a 64-bit block cipher which was intended to be an attractive alternative to DES[22].

The following lines of codes are the encryption algorithm used in the Your Right voting system:

```
functionpass_encrypt($pass){
  $hash_format = "$2y$10$$";
Calling function generate salt:
   $salt = generate_salt();
   $format_and_salt = $hash_format.$salt;
   $hash = crypt($pass, $format_and_salt);
         return $hash;
}
```

Generating salt function:

```
functiongenerate_salt(){
   $uni_rand= sha1(uniqid(mt_rand()), true);
   $base64_string  = base64_encode($uni_rand);
   $mod = str_replace('+', '-', $base64_string);
   $salt = substr($mod, 0, 22);
return $salt;
}
```

Table No.1: Html character encoding

S.No	Convert	To
1	&	&amp
2	<	&lt
3	>	&gt
4	"	&quot
5	'	&#x27

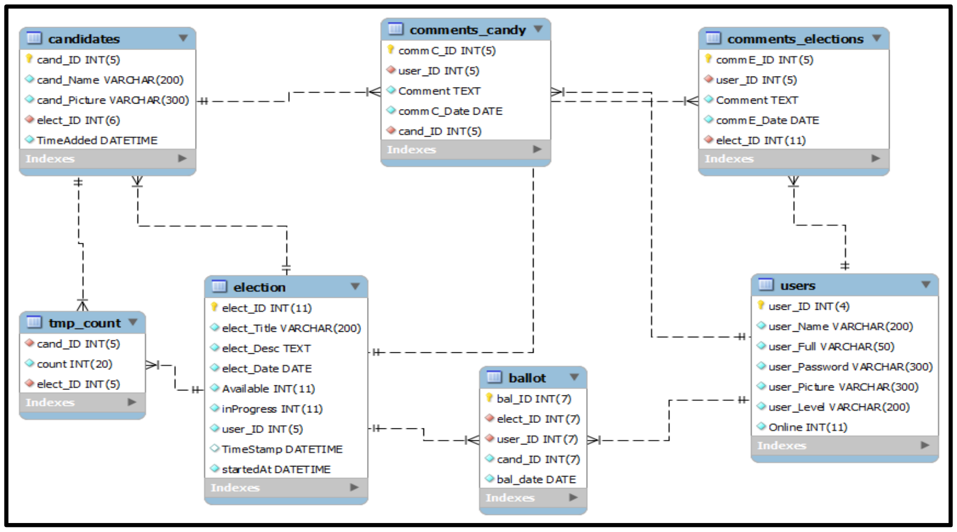


Figure No.1: The entity relationship diagram of Your Right

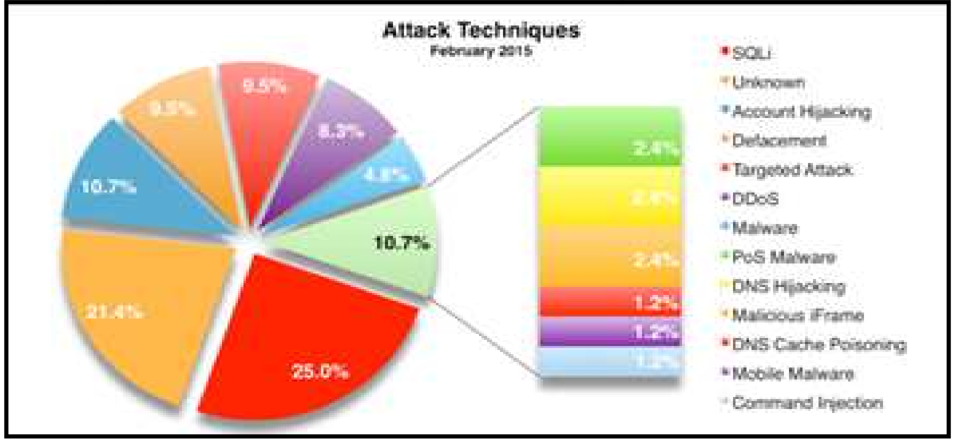


Figure No.2: February 2015 Cyber Attacks Statistics

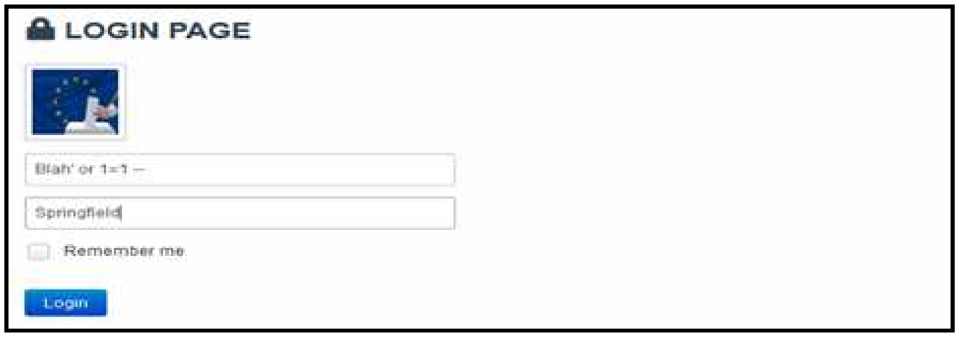


Figure No.3: This figure shows an attacker launching an SQL injection on Your Right electronic voting system

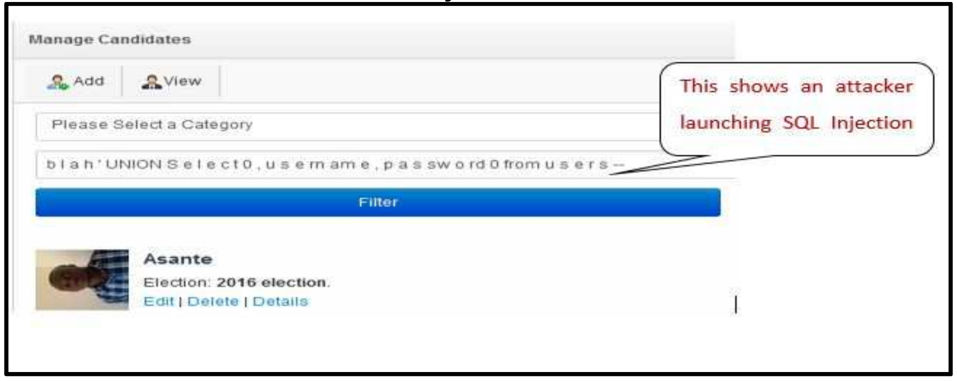
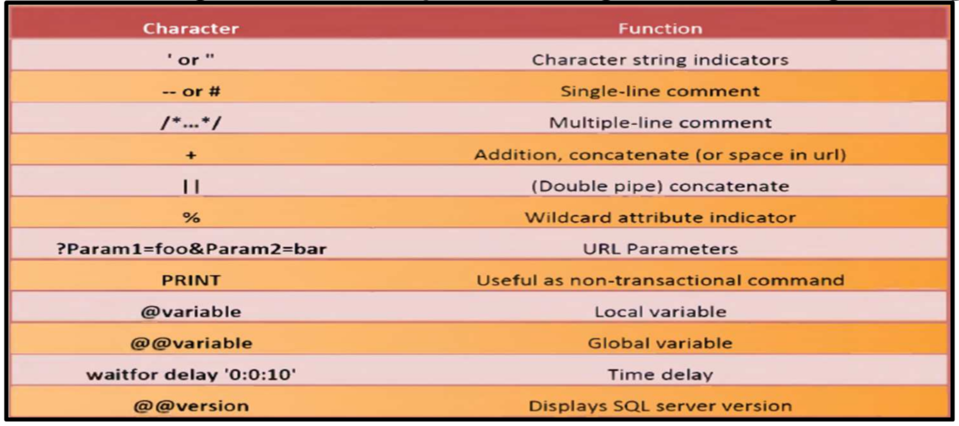


Figure No.4: This image shows SQL Injection testing on the Your Right voting system



Character	Function
' or "	Character string indicators
-- or #	Single-line comment
/*...*/	Multiple-line comment
+	Addition, concatenate (or space in url)
||	(Double pipe) concatenate
%	Wildcard attribute indicator
?Param1=foo&Param2=bar	URL Parameters
PRINT	Useful as non-transactional command
@variable	Local variable
@@variable	Global variable
waitfor delay '0:0:10'	Time delay
@@version	Displays SQL server version

Figure No.5: SQL Injection Attack Characters

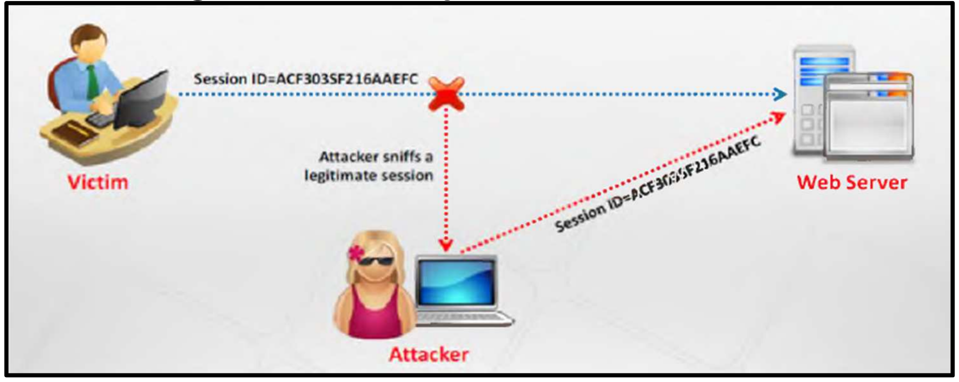

Session ID=ACF303SF216AAEFC
Victim
Attacker sniffs a legitimate session
Session ID=ACF303SF216AAEFC
Web Server
Attacker

Figure No.6: diagrammatical representation of an attacker sniffing a session securing Your Right voting system

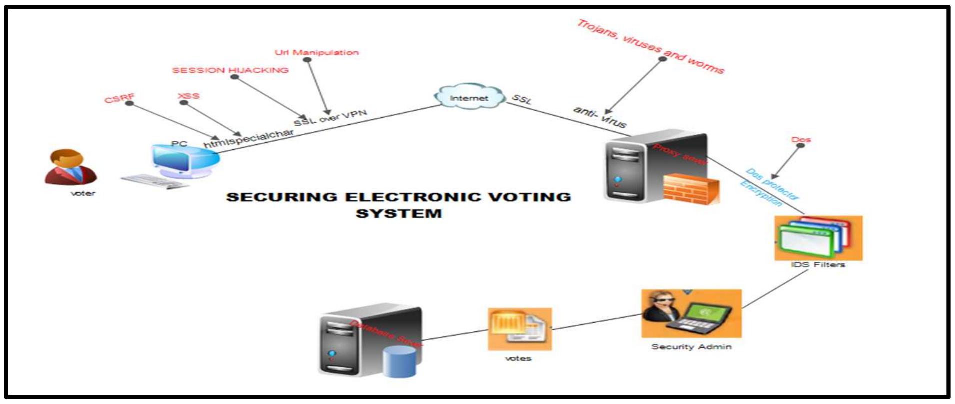


Figure No.7: Diagrammatical representation of securing Your Right voting system

CONCLUSION

Web applications have be predominantly common in our world today and so as web security issues have become a problem for web developers, network administrators, and security analysts. Attackers spend most their times to look for web applications that are susceptible to vulnerabilities like SQL Injection, Cross-Site Scripting, and Session Hijacking. When these attackers find vulnerable systems or web applications, they begin to attack them with the intention of taking over these systems or being administrators of these systems. They do so by gain unauthorised access to these systems. The most common systems that have come under scrutiny are the electronic voting systems that are used during elections.

There are many vulnerabilities that are associated with electronic voting systems of which Your Right voting system is included. In this paper, we have looked at vulnerabilities like SQL Injection, session hijacking, Cross-Site Scripting and their countermeasures. We have also looked at an encryption algorithm to encrypt data in the Your Right voting system.

Finally, a diagram was designed to show how to secure Your Right voting system to safeguard our electoral integrity and to make voters trust electronic voting.

ACKNOWLEDGEMENT

The authors wish to express their sincere gratitude to Department of Information Technology, Valley View University, Ghana, West Africa for providing necessary facilities to carry out this research work.

CONFLICT OF INTEREST

We declare that we have no conflict of interest.

BIBLIOGRAPHY

1. Mark D Phillips, R W S. “Testing Democracy: How Independent Testing of E-Voting Systems Safeguards Electoral Integrity,” *Bib Sonomy*, 205 of LNI, 2012, 159-170.
2. Oostveen, A. a. P. v. d. B. "Electronic Voting in Europe: Technology, Law, Politics and Society.," In *Security as Belief.* Users Perceptions on the Security of Electronic Voting Systems, Bonn, *Gesellschaft fur Informatik*, 47(1), 2004, 73-82.
3. Chin-Ling Chen, Y.-Y. C. J.-K. J. C.-C. C. "A Secure Anonymous E-Voting System based on discreet logarithm Problem," *Applied Mathematics and Information Sciences*, 8(5), 2014, 2571-2578.
4. Jaydeep Howlader, V. N. S. B. a. A. K. M. "Uncoercibility in e-voting and e-auctioning mechanisms using deniable encryption," *International Journal of Network Security and Its Applications (IJNSA)*, 3(2), 2011, 97-108.
5. Niranjanamurthy M, D. D. C. "The study of E-Commerce Security Issues and," *International Journal of Advanced Research in Computer and Communication Engineering*, 2(7), 2013, 2885-2895.
6. Razibul Hasan M A S. "Study on e-Commerce Threats and Security," *National Conference on Communication and*

Information Security (NCCIS 2012), 3(12), 2012, 76-83.

7. Sengupta A, C. M. a. M. S. B. "e-Commerce security - A life cycle approach," *Sadhana,* 30(2), 2005, 119-140.
8. Olayemi Mikail Olaniyi, O. T. A. a. E. O. O. "Design of Secure Electronic Voting System Using Multifactor Authentication and Cryptographic Hash Functions," *International Journal of Computer and Information Technology,* 2(6), 2013, 2279-0764.
9. Saman Shojae Chaeikar, M. J. H. T. N. S. C. K. "Definitions and Criteria of CIA Security," *International Journal of Advanced Computer Science and Information Technology,* 1(1), 2012, 14-24.
10. Peralta R. "Issues, Non-Issues, and Cryptographic Tools for Internet-Based Voting," in *Secure Electronic Voting, US, Springer,* 7(1), 2003, 153-164.
11. Qing Li, P. Y.-L. C. "Entity-Relationship Diagram," in Modeling and Analysis of Enterprise and Information Systems, *Berlin Heidelberg, Springer,*), 2009, 125-139.
12. Satapathy, S. G. A. R. K. M. J. "SQL Injection Detection and Correction Using Machine Learning Techniques," in Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India (CSI) Switzerland, *Springer International Publishing,* 1(3), 2015, 435-442.
13. Passeri P. "hackmageddon.com," 9 March 2015. [Online]. Available: http://hackmageddon.com/. [Accessed 24 March 2015].
14. EC-Council, " SQL Injection," in Ethical Hacking and Countermeasures Version 8, *EC-Council press*, 2013, 1987-2133.
15. Johns M. "Session Hijacking Attacks," in *Encyclopedia of Cryptography and Security, US, springer*, , 2011, 1189-1190.
16. EC-Council, "Session Hijacking," in Ethical Hacking and Countermeasurs Version 8, *EC-Council*, 2013, 1504-1599.
17. Balakrishnan. PHP Login Script, 2010.
18. Ashar Javed J R S. "SIACHEN: A Fine-Grained Policy Language for the Mitigation of Cross-Site Scripting Attacks," in Information Security, Switzerland, *Springer International Publishing*, 5(3), 2014, 515-528.
19. owasp, "owasp," owasp, 22 04 2014. [Online]. Available: https://www.owasp.org /index.php/Cross-site_Scripting_(XSS). [Accessed 6 April 2015].
20. Chris Snyder M S. "Preventing Cross-Site Scripting," in *Pro PHP Security, Apress,* 1st edition, 2005, 263-279.
21. owasp, "www.owasp.org," owasp, 2 03 2015. [Online]. Available: https://www.owasp.org /index.php/XSS_%28Cross_Site_Scripting% 29_Prevention_Cheat_Sheet. [Accessed 6 April 2015].
22. EC-Council, "Cryptography," in Ethical Hacking and Countermeasures V8, *EC-Council Press*, 2013, 2783-2871.
23. Canniere C D. "Blowfish," in *Encyclopedia of Cryptography and Security, US, Springer US*, 3(3), 2005, 48-49.

Please cite this article in press as: Edward Danso Ansong *et al*. An electronic voting system to seguard electoral integrity, *International Journal of Engineering and Robot Technology,* 3(2), 2016, 39-47.