International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Architectural Design of a Secured Web-Based Healthcare Portal

Rotimi-Williams Bello
Department of Mathematical Sciences, University of Africa, Toru-Orua, Bayelsa State, Nigeria
Email address: sirbrw@yahoo.com

Abstract*—Web-based healthcare portal refers to the health-related services that are accessed over a network connection using Hyper Text Transfer Protocol (HTTP), rather than existing within a device's memory. However, Web-based healthcare portal also may be client-based, where a small part of the health-related services are downloaded to a user's desktop, but processing is done over the internet on an external server through which the portal is exposed to a lot of security pitfalls. Therefore, this research work intended to design defense architecture with security check point to detect the Web-request-level attacks by employing request blocker as the security check point. The findings in this research work showed that the Web-based healthcare portal controls the access over its functions by checking session variables indicating the user privilege before its restrictive functions can be executed. If the application is not at the required state, the Web-based healthcare portal will redirect the user to the login page, authorization page or an error page. However, if there exists a path leading to the restrictive function with insufficient or erroneous checking of session variables, the attacker is able to bypass the authentication/authorization*.

***Keywords*—** *Blocker; Portal; Network; Hyper Text Transfer Protocol; and Security*.

I. INTRODUCTION

The patient portal is an essential link in the chain of ensuring patient data security and privacy. In August 2000, it was reported that over 800 patients' sensitive information was breached through KPOnline, a Web-based healthcare portal, due to a small piece of flawed software that was integrated into the system [1]. Developing a secured Web-based healthcare portal is a complex and challenging task because the development adds additional layer of complexity during implementation. The healthcare portal has to implement and enforce complex security policies to restrict the access to sensitive information and actions such as diagnosis code and secret tokens against common security pitfalls, weak authentication, cross-site scripting, and SQL injection. This complexity is increased because of the concurrent nature of activities on the internet, and the interplay among many users. Such policies are usually dynamic, related with clinical workflows, thus cannot be explicitly or precisely defined. Therefore, in this research work, blocker approach was used as improved methods for detecting and blocking Web application attacks. World Health Organization (WHO) defines e-health as the cost-effective and secure use of Information Technology (IT) in support of health and health-related fields, including healthcare services, health surveillance, health literature, and health education, knowledge and research [2]. E-health is the use of IT for health. E-health then encompasses services such as health-related internet information sites, automated online therapy, email consultations, online pharmacies, telehealth, home monitoring systems and virtual clinics. It also includes information technology-based health system developments. It comprises health promotion, disease prevention and care to improve health conditions and equity. Involving different stakeholders with different interests and needs, this requires a plurality of solutions in meaningful contexts. But, it is an unfortunate reality that healthcare is not as safe as it should be. Adverse events and preventable errors that can cause harm and death are commonplace in healthcare. These errors are most often not the fault of individuals, but of a system that fails to provide safe and effective care. The cause of preventable errors can be traced to gaps in the flow of information and communication failures both within organizations and across different healthcare service providers. The personal cost of these errors is immeasurable. IT is therefore used to describe a range of technologies for gathering, storing, retrieving, processing, analyzing, and transmitting information, as information is seen as a key element and workforce trained in the appropriate health information skills [2]. IT has helped in bridging distances and providing access to clinical knowledge, specialized expertise and healthcare services thus saving lives and costs. IT provides access to clinical information, telemedicine, online discussion groups and other tools. The need for reform of health sector and the need for investment in, and deployment of e-health has been part of the healthcare agenda for many years [2]. Sisniega asserted that the applications of ITs facilitate ubiquitous and instantaneous communication between organizations and their stakeholders and enables people and organizations to achieve a seamless workflow and effective processes through improved interactions [3]. The available literature provides a common position among various authors that disparities exist in the implementation of healthcare systems in developing and developed countries [4], [5]. Speculated reasons include poor technological and funding support in developing nations, poor management capacity at all levels that ensures seamless workflow, and a complex milieu of healthcare service delivery. Other possible factors for low implementation include the continual evolution of technology, confidentiality problems with use of hospital information systems, and the poor technological background of the society [4], [6]. Holden posited that much research related to adoption of healthcare information technology has been a theoretical [7]. A useful theoretical model is the maturity model to process improvement originally described

Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

in software engineering and used in the novice-to-expert approach to competency. The maturity theoretical model describes a modernization framework aimed at the committed use of relevant information technology in a change process [8]. Beneficial uses of information and associated technology as it relates to healthcare improvement in this model includes monitoring individual and organizational performance, facilitating information sharing among different healthcare organizations through a multi-agency approach, and empowering individuals by providing relevant information to consumers, thereby helping them to make informed choices [8]. Therefore, there is a rising demand for healthcare services due to the ageing of the population, the rise in chronic disease and increased consumer expectations; problems with health workforce supply and distribution; inequity of access to services, particularly amongst indigenous, rural and poor populations; quality and safety concerns; and fragmented and limited ability to share information [9]. The extent of the challenges is great and they must be met with less resource [10]. Web-based system, a product of IT is a program that is accessed over a network connection using hypertext transmission protocol (HTTP), rather than existing within a device's memory [10]. Web-based applications often run inside a Web browser. However, Web-based applications may be client-based, where a small part of the program is downloaded to a user's desktop, but processing is done over the Internet on an external server. Web-based healthcare portal on the other hand is healthcare related online applications that allow patients to interact and communicate with their healthcare providers, such as physicians and hospitals. Typically, portal services are available on the internet at all hours of the day. Some patient portal applications exist as stand-alone Web sites and sell their services to healthcare providers. Other portal applications are integrated into the existing Web site of a healthcare provider. Still others are modules added onto an existing electronic medical record (EMR) system. What all of these services share is the ability of patients to interact with their medical information via the Internet. In Nigeria for example, healthcare providers managing the health conditions of patients have independent medical records systems. The result is a multiplicity of parallel medical records systems on the same population. Integrating these systems, so that specific and selected patient records can be shared is desired but has not been attained as yet. Healthcare managers, supporting staff and their patients alike continue to be denied quick access to information they require for the successful fight against ailments. The development of Web systems (of which Web sites are the most common example) is an important task in modern software engineering [11]. This task is complicated by many factors. One of those factors is the possibility of a concurrent access to a Web system by many users. If this concurrency is not managed correctly, the data in those systems may become inconsistent. Another factor, related predominantly to Web sites, is related to inherent problems of Web-browsers. For example, accessing a Web system from multiple browser windows may lead to incorrect behavior of a system [12]. Also, most of the Web systems require that access to certain resources is only available to some users, i.e. the Web systems should be secured. A protection of this kind includes application of methods for authentication and authorization [13]. A lack of this kind of protection may lead to serious economic and social consequences. So, an additional theoretical standpoint in this study is that in a heterogeneous society as Nigeria with significant disparity in accessibility of healthcare facilities between urban and rural communities, Web-based healthcare portal need to be security oriented to help bridge the gap in availability of patient care with the protection and confidentiality of such patient related data and information [14]. Secured Web-based healthcare portal; an online service is a server based approach to protect sensitive data and information when transacted between the healthcare providers and their patients and provides compliance with industry regulations and it provides non-repudiation as the recipients (patients) are personally identified and transactions are logged by the secured platform. Patients are enrolled with their medical records to a secured transaction platform and access to the records is by providing username and password (or strong authentication).

II. METHODOLOGY

In this section, diagrams of a secured Web-based healthcare portal with security check point are presented. Because the most common internet attack is Web-request-level attacks [15-18] therefore, security of Web-based healthcare portal request-level can be specifically done through two approaches [19]; one consisting of rules that represent security policies in a deterministic manner (e.g., access control list), and the other including evidence specifications with statistical measures to effectively handle the complex and dynamic policies that are unique to the clinical environment, which are usually not predefined, or, in some cases, unable to be defined accurately. The thesis approach to security of Web-based healthcare portal request-level is to use rule-based specification on which request blocker that operates over a central decision engine is based because in a healthcare portal that is Web-based, all electronic medical records about patient can be found in the database with series of tables. A request is therefore consists of operations on tables. The security specification is inferred from the interactions between the patient portal, Web clients, electronic medical record (EMR) database and other components. The inferred security specifications capture the intended behaviour of the Web-based healthcare portal and are verified and associated with clinical semantics by security experts to suppress false positives. As earlier iterated, the research work employed request-blocker as security-check-point. The request-blocker, which sits between the users and the healthcare portal, operates over a central decision mechanism based on repository of security specifications (rules), and from the interactions between the healthcare portal, Web clients, EMR database and other components, the security specifications (rules) are inferred. The decision mechanism has access to the user session information through connectors which can read local session files or retrieve information from database table; this is capable of evaluating

Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

the request and queries in a context-aware manner and identifying logic attacks.

A. Architecture of a Secured Web-Based Healthcare Portal With Security Check Point

In this architecture, the security specification inferred by the inference engine is used to evaluate each incoming Web request and outgoing Web response and detects any violations. As shown in Fig. 2.1, blocker monitors the interactions between the clients and the Web-based healthcare portal, dynamically detects and blocks any potential attacks.

B. Components of the Architecture

a. User: This is the person that has the right to access Web-based healthcare portal. Most of the times, he/she is either the record officer or the patient's healthcare provider.
b. Web request: This is the act of inquiring for information on the Web by submitting query.
c. Template: A model, standard or outline to follow or adapt in Web-based applications.
d. Invariant: A feature (quantity, property or function) that remains unchanged when a particular transformation is applied to it.
e. Blocker: Presented in Fig. 2.1 and Fig. 2.2 are request blockers, sitting between the user and the Web-based healthcare portal to detect the Web-request-level attacks and prevents sensitive information to be revealed to the attackers in Web responses. Blocker approach is based on the BLOCK system of Li and Xue [20]. The approach to security of Web-based healthcare portal request-level is to use rule-based specification on which request blocker that operates over a central decision engine is based.

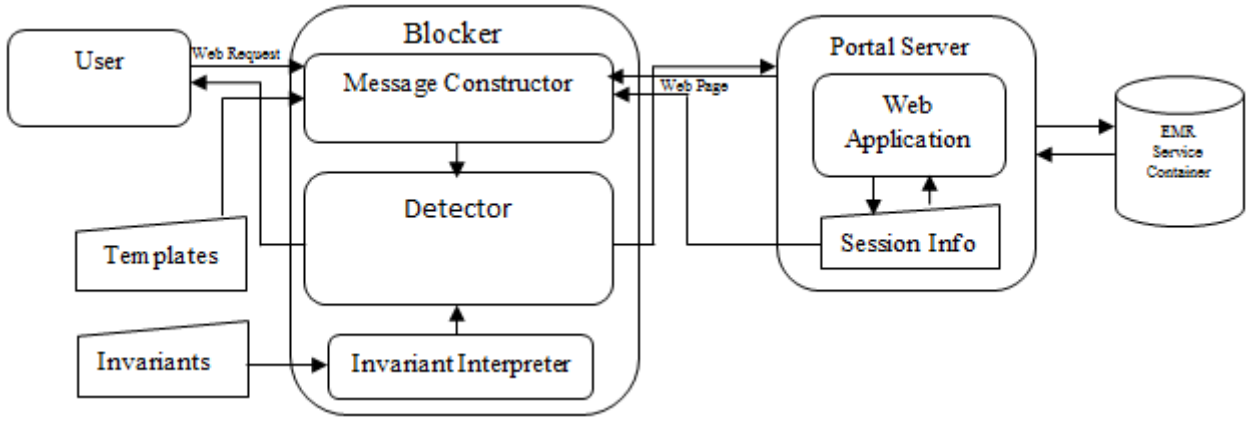

Fig. 2.1. Architecture of a Secured Web-Based Healthcare Portal with Security Check Point.

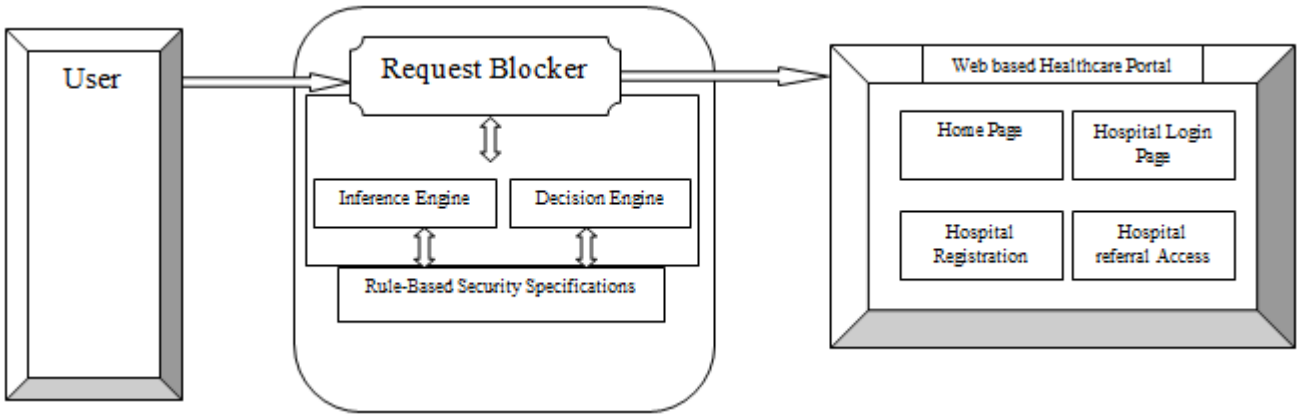

Fig. 2.2. Components of Request Blocker.

i. Inference Engine: The security specification is inferred by the inference engine from the interactions between the patient portal, Web clients, EMR database and other components. The inferred security specifications capture the intended behaviour of the Web-based healthcare portal [21].
ii. Decision/Access Control Engine: Presented in Fig. 2.3 is decision/access control engine; this is the engine over which request blocker operates for the security of Web-based healthcare portal. The decision/access control engine is a dedicated service that performs rule-based access control for users' requests. It contains the following components:

- Evaluator: The core of the engine is an evaluator, which acts as a reasoning system to validate the rule set. This component is written in SQL and runs in the SQL server.

- Predicates API: Another important component of the engine is the predicates API, which implements all required user-specific predicates used in the system. It is also responsible for collecting data, such as user profile, appointment/referral repository, current server time etc., to instantiate the variables in each predicate. This component is written in VB.NET.
- Interface: VB.NET application is another component that sends request to and receives response from the SQL server and exposes the validation interfaces as a Web service which can be invoked by access control proxies resided in applications. Each business policy can be represented as a rule set, which contains multiple rules and meta-policies. The VB.NET application hides the rule validation details and only exposes one interface with few parameters. When the access control proxy wants to validates user's request, it only needs to provide the necessary values.

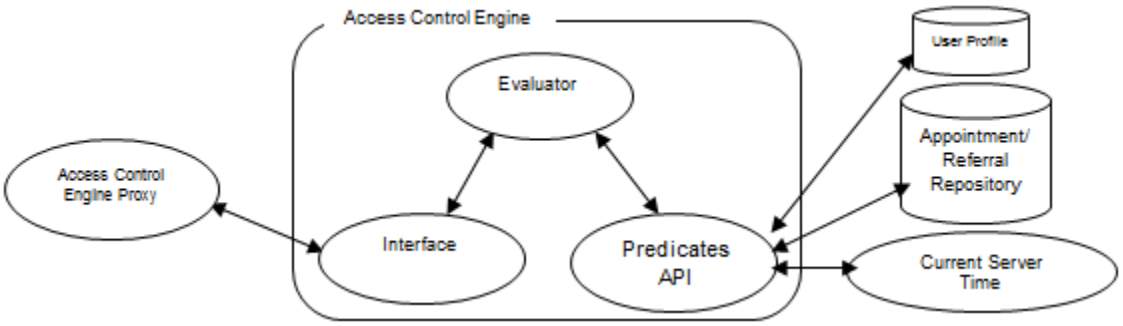

Fig. 2.3. Components of Access Control Engine.

ii. Rule-based security specification: This is where request blocker that operates over a central decision engine is based and it consists of rules that represent security policies in a deterministic manner (e.g., access control list). The security specification is inferred from the interactions between the patient portal, Web clients, EMR database and other components. The inferred security specifications capture the intended behaviour of the Web-based healthcare portal and are verified and associated with clinical semantics by security experts to suppress false positives. Security specifications, maintained by decision engine, are learnt by an inference engine from real Web traffic that reflects dynamic and complex security policies in clinical environment. The following are the portal security rules against Web-request-level attacks:

```
<?asp
include_once("header.asp");
if (isset($_GET['logout'])){
        session_start();
        unset($_SESSION['username']);
        unset($_SESSION['privilege']);
        session_destroy();
        print "You are logged out.<br>";
} else if (isset($_POST['email'])){
        if (validateLogin($_POST['email'],
$_POST['passwd'])){
                $_SESSION['username'] = $_POST['email'];
                if ($_POST['email'] == $admin_email){
                        $_SESSION['privilege'] = "admin";
                } else {
                        $_SESSION['privilege'] = "user";
                }
                header("Location: index.asp?username
                        =" . $_SESSION['username']);
                        exit();
                } else {
                        die("Wrong username or
                password");
                }
}
?>
<form action='login.asp' method=post>
username: <input name="email" type="text"><br>
password: <input name="passwd" type="password"><br>
<input name="submit" type="submit"></form>
<?asp include_once 'footer.html';?>
```

login.asp

```
<?asp include_once 'header.asp';
logIdentity();
print "<a href='admin2.asp'>Next step: change the
title</a>";
include 'footer.html'; ?>
```

admin.asp

```
<?asp
include_once 'header.asp';
if (isset($_GET['username'])){
        $userid = $_GET['username'];
        showUserInfo($userid);
        if ($_SESSION['privilege'] == "admin"){
                print "<a href='admin.asp'>Admin
        link</a><br >";
        }
}
print "<a href='login.asp?logout=1'>Logout</a><br>";
include_once 'footer.html';
?>
```


Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

index.asp

```
<?asp
include_once 'header.asp';
if ($privilege != "admin"){
        header("Location: index.asp?username
                ="."$_SESSION['username']);
}
if (isset($_POST['title'])){
        modifyTitle($_POST['title']);
}
?>
<form action='admin2.asp' method=post>
New title: <input name="title" type="text"><br>
<input name="submit" type="submit">
</form>
<a href='login.asp?logout=1'>Logout</a>
<?asp
include_once 'footer.html';?>
```

admin2.asp

Under the portal security rules, the Web-based healthcare portal controls the access over its functions by checking session variables indicating the user privilege before its restrictive functions can be executed. If the application is not at the required state, the Web-based healthcare portal will redirect the user to the login page, authorization page or an error page. However, if there exists a path leading to the restrictive function with insufficient or erroneous checking of session variables, the attacker is able to bypass the authentication/authorization. Portal security rules demonstrate three cases of auth bypass attacks: *admin.asp and admin2.asp* contain restrictive functions, which should only be accessed by admin users when the session variable $_ *SESSION ['privilege']* is set to the value of *admin.*

i. *In admin.asp,* there is no check on the session variable *$_SESSION ['privilege']*. The attacker, being either a guest or a regular user, can directly request the page and access the admin functions.
ii. *In admin2.asp*, even though there is an if condition check on the session variable *$privilege*, the attacker can append an additional parameter privilege to the URL, for example http://example.com/admin2.asp*? privilege=admin*, and bypass the auth check. The reason is when the register global option of ASP interpreter is enabled, the parameter attached to the Web request will be automatically bound to a global variable, if such variable doesn't exist in the current session state. This vulnerability results from the inappropriate or erroneous check on the session variable.
iii. *In admin2.asp*, even when the auth check fails, the attacker is able to execute the restrictive functions after the redirection (i.e., header function) by submitting a POST request with the parameter title and change the application's title successfully. This is because there is no exit function or an additional check after the redirection.

In a lot of cases, the Web-based healthcare portal assumes implicit relations between the user's input parameters within Web requests and the session state. Such a relationship may also be reflected from Web responses returned by the Web-based healthcare portal. If the application doesn't check the session state when accepting the Web request, the attacker is able to manipulate the input parameters and gain access to unauthorized information. In portal security rules, after the user logs in, he/she will be redirected to the *index. asp page,* which displays his/her personal information. The Web application assumes the request parameter *username* is always equal to the value of session variable $_SESSION *['username']*. If the equality relationship is not examined when the user's personal information is retrieved, the attacker is able to view any user's information by modifying the username parameter within the Web request. Also, a Web-based healthcare portal usually has an intended workflow, which requires the user to perform a predefined sequence of operations to complete a certain task. For example, an e-commerce Website has a predefined checkout procedure, which instructs the customer to first fill in the shipping information and then the credit card information before the order can be confirmed and submitted [21]. Such a temporal relationship is enforced by the restrictions over the session state transitions. However, if the session variables are insufficiently defined or checked for guarding the desired state transitions, the attacker is able to bypass certain required steps and violate the intended workflow. Figure 3.4 requires the admin user first access *admin.asp,* which logs his/her identify (by *logIdentity* function) before he/she can modify the application title in *admin2.asp.* The two steps indicate two different session states and the transition between them should be guarded by the Web application. However, there is no session variable defined for indicating whether the identity of the admin user has been logged or not. The attacker can directly point to *admin2.asp* page without his/her identity being logged.

f. *Message Constructor*: At runtime, the message constructor combines session information with the intercepted Web request, composes an input and sends it to the detector for evaluation. If the input is accepted, the Web request is forwarded to the Web-based healthcare portal and logged as the current input for the Web-based healthcare portal. Otherwise, the Web request is dropped. When the message constructor receives a Web response, if the response is a redirection, the subsequent Web request will not be evaluated or logged. If the response is a Web page, the message constructor assigns the Web page a response key based on its Web template, composes an output and sends it to the detector, where the output is paired with the current input and evaluated. If the output is accepted, the Web page is returned to the client and the key pair is logged for the current user session. Otherwise, the Web response is blocked and the current input is invalidated. After the user's session has terminated, all of the logged key pairs are cleaned up.
g. *Detector:* This is the component that is used to evaluate the composed input and output received from constructor.
h. *The invariant interpreter*: The invariant interpreter loads and interprets the extracted invariants.

Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

i. *Web page*: A document (medical records) connected to the World Wide Web and viewable by anyone connected to the internet that has a Web browser and the access right.
j. *Portal server*: A computer on the internet positioned by the site owner (healthcare provider) as an entrance to other sites to provide client stations with access to files and printers as shared resources to a computer network.
k. *Web application*: Computer network consisting of a collection of internet sites (healthcare portals) that offer healthcare services and medical records information through the hypertext transfer protocol.
l. *Session info*: This takes into account the information about the session state of a Web request and response.
m. *EMR service container*: This is acronym for Electronic Medical Records service container. It is the Web-based container that keeps record information about all the activities of healthcare providers and their patients.

C. Security Model Description Of The Blocker Functionality

As presented in Fig. 2.1, the thesis approach follows the stateless property of HTTP and regards the session variables as part of the input to the Web-based healthcare portal along with Web requests. Similarly, the output of the application consists of the Web response and the session variables. In this way, the Web-based healthcare portal can be regarded as a stateless system, as shown in Fig. 2.4. Under this stateless system model, the application behaviour is characterized in the form of three types of likely extracted invariants.

a. Type I input invariants, indexed by the Web request key *r*: For the fact that portal input consists of the Web request and the values of the session variables when the request is made, this type of invariants models the relationship between the Web requests and the session variable values. Essentially, it tries to capture the constraints on the Web requests at certain session states. By identifying the invariant component of session variables, this approach avoids the introduction of spurious states by unnecessary session variables. The inputs with the same request key *r*are grouped together. The following types of invariants for each request key *r* are extracted.
   i. A set of session variables $S_{inv}(r)$ that is always present. An example invariant of this type is S_{inv}*(GET-index.asp) = {$_SESSION['username'], $_SESSION['privilege']}.*
   ii. A set of input parameters $P_{inv}(r)$ that is always present. An example invariant of this type is P_{inv}*(POSTlogin.asp)* = {email, passwd}.
   iii. For a specific session variable $s \in S_{inv}(r)$, its value is drawn from an enumeration set $V(s, r)$. For example, invariants of this type include: *V ($_SESSION ['privilege'], GETadmin. asp)* = {admin}, *V ($_SESSION ['privilege'], GETindex. asp)* = {admin, user};
   iv. For a specific input parameter $p \in P_{inv}(r)$, its value is drawn from an enumeration set $V(p, r)$.
   v. The value of an input parameter $p \in P_{inv}(r)$ is always equal to the value of a session variable $s \in S_{inv}(r)$. For the request key *GET-index.asp*, the session variable *$ SESSION ['username']* is always equal to the input parameter username.

b. Type II input/output invariants, indexed by the key pair (*r, v*): This type of invariants models the relationship between the Web request and response as well as the changes in the session variables after the Web request is processed. Essentially, it tries to capture the constraints on the application state transition and the input/output dependency at a certain state. Both type I and II invariants rely on the session variables to infer the application states. When the session variables are not sufficiently defined, a third type of invariant is needed. The input/output pairs with the same key pair (*r, v)* are grouped together. The same set of invariants as type I is first extracted for the key pair. For example, an invariant drawn for the key pair *(GET-login.asp, t.logout)* is that *V(logout, (GET-login.asp, t.logout)* = {1} and the input parameter logout is added into P_{inv}*(GET-login.asp, t.logout)*. Two new invariants for each key pair (*r, v*) are also extracted:
   i. The value of an output parameter is always equal to the value of an input parameter and/or a session variable. This invariant reflects the dataflow within the Web application. An invariant for the key pair *(POST-login.asp, t.index user)* is that the output parameter /html/body/ of the template *t*.index user is always equal to the session variable *$_ SESSION['username']* and the input parameter username.
   ii. The session state is unchanged. For example, the user's session state always stays the same by observing the key pair *(GET-login.asp, t.login form)*, but evolves for the key pair *(POST-login.asp, t.index user)*.

c. Type III input/output sequence invariants, also indexed by the request key *r*: This type of invariants models the relationship between consecutive Web request/response pairs. Essentially, it tries to capture the application states that are not revealed by defined session variables by leveraging the historical request/response information. For each request key *r*, the following invariant are extracted:
   i. A set of input/output key pairs that always precede the Web request key in one session. An invariant of this type is the key pair *(GET-admin.asp, t.admin)* always precedes the request key *GET-admin2.asp* and the key pair *(GET-admin2.asp, t.title form)* always occurs before *POSTadmin2.asp*.

D. Portal Database Encryption

For Web request-level-attacks detection, each invariant is transformed into an evaluation function as shown in Fig. 2.5, Fig. 2.6, Fig. 2.7, and Fig. 2.8 respectively, which operates on an input or an input/output pair. If the input or input/output pair satisfies the invariant, the function returns true. Otherwise, the function returns false. The runtime detection is performed in two phases:

a. Validating the input: The Web request is accepted, if and only if the request key has been observed and all the invariants associated with it are satisfied. Otherwise, the Web request is dropped.

Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

b. Validating the input/output pair: The Web page is sent back to the user if and only if the corresponding key pair has been observed and all the invariants associated with it are satisfied. Otherwise, the Web page is blocked.

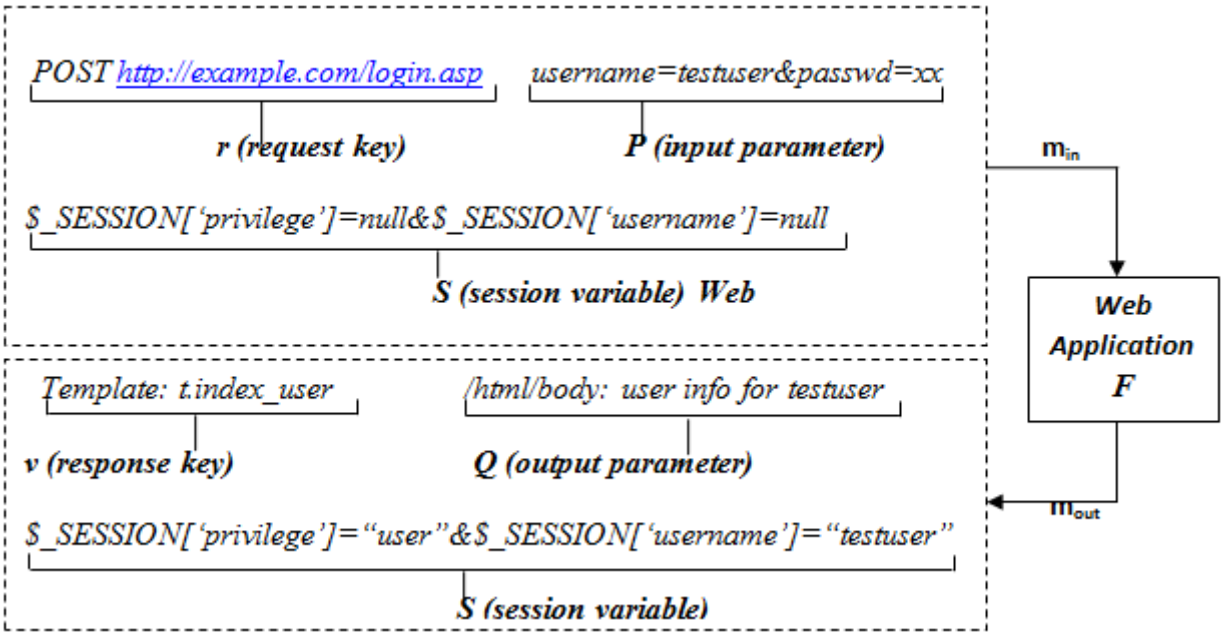

Fig. 2.4. A Stateless View of Web-Based Healthcare Portal.

Fig. 2.5. User Access Table.

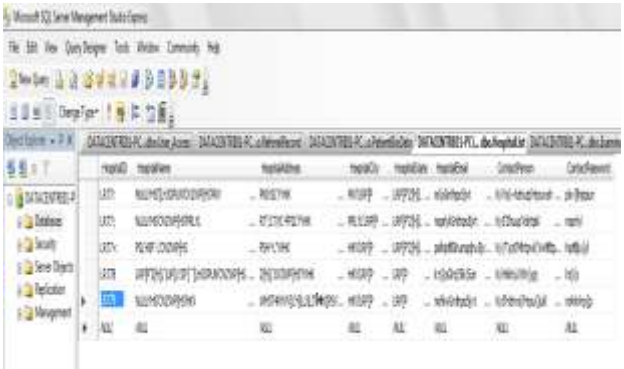

Fig. 2.6. Hospital List Table.

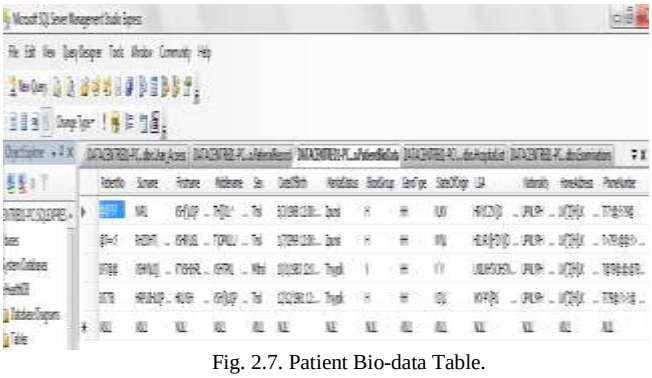

Fig. 2.7. Patient Bio-data Table.

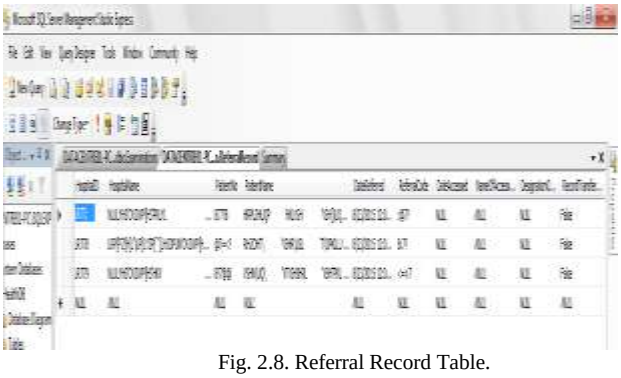

Fig. 2.8. Referral Record Table.

All the intending attacks can be detected when:

a. Each auth bypass attack instance violates the invariants associated with request keys and is detected.
b. The parameter manipulation attack violates the invariant associated with the request key where the input parameter username is always equal to the session variable and is detected.
c. The workflow bypass attack violates the invariant associated with the request key that the key pair always precedes the request key and is detected

Rotimi-Williams Bello, "Architectural design of a secured web-based healthcare portal," *International Research Journal of Advanced Engineering and Science*, Volume 3, Issue 2, pp. 332-340, 2018.

III. RESULTS

In order to find the effectiveness of the approach used in this research work, some experiments were conducted on the source codes. In the experiment, different illegal request attempts were made on the Web applications especially, on the patient referral record (medical record) as shown in Fig 3.1.

Fig. 3.1. Illegal Attempt on Web-Based Patient Referral Record.

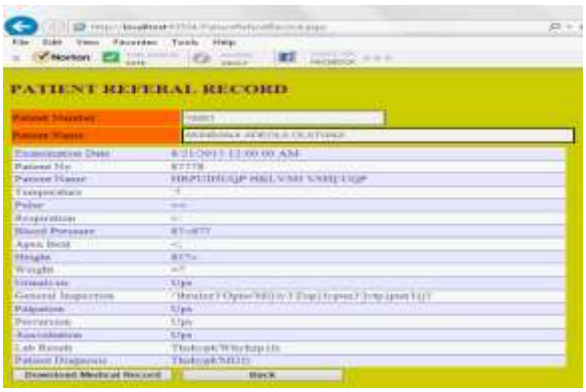

Fig. 3.2. Encrypted Download of Patient Medical Record.

The download medical record was a complete encrypted record as shown in Fig 3.2 making it impossible for attackers to get decrypted records.

IV. CONCLUSION

The security architecture approach used in this research work is valuable in that it is independent of the Web application source code and suitable for a large variety of Web application hosting scenarios based on different application frameworks, where the source code may not be available. Visual Basic.NET, the programming language that was used for the implementation possesses some characteristics and capability that underline the decision to use it. The rich and extensive API enables developers to easily incorporate packages into their codes which enhance reusability and utility of codes. The following are some of the additional advantages of the programming language used in the thesis:

a. Code can be created faster and better because there is no need to create functionality from scratch. Best of all, many developers can create their own packages and make them open-source, allowing other developers to use their codes and hopefully make improvement.
b. The packages are well documented, so learning about a particular method is as easy as incorporating them in one's code, a phenomenon that dramatically improves programmer's productivity.
c. The major problems in programs like memory management and bounds checking are not seen in VB.NET, all these issues are handled automatically.
d. Serial I/O, networking, graphics sound, and even video are all supported packages.
e. Most importantly, administrators' passwords and other sensitive details resident in database are well encrypted against manipulations.

Also, many sites from the last decade are static, but more and more people are realizing the advantages of having a dynamic portal, among which are: (1) it possesses some security features, (2) it is much easier to update, (3) it is much more functional Website, (4) its new content brings people back to the site and helps in the search engines, (5) it can work as a system to allow staff or users to collaborate. Although, to develop a secured Web-based healthcare portal is a complex and challenging task because the development adds additional layer of complexity during implementation. The healthcare portal has to implement and enforce complex security policies to restrict the access of sensitive information and actions such as diagnosis code and secret tokens against common security pitfalls. This complexity is increased because of the concurrent nature of the internet, and the interplay among many users. Such policies are usually dynamic, related with clinical workflows, thus cannot be explicitly or precisely defined.

REFERENCES

[1] Collmann, J. and Cooper, T., "Breaching the security of the Kaiser Permanente internet patient portal: The organizational foundations of information security," *Journal of American Medical Informatics*, vol. 14, pp. 239–243, 2007.
[2] Berland, G. K., Elliott, M. N. and Morales, L. S., "Health information on the internet: Accessibility, quality, and readability in English and Spanish," *Journal of the American Medical Association*, vol. 285, issue 20, pp. 2612-2621, 2001.
[3] Sisniega, L. C., "Barriers to electronic government use as perceived by citizens at the municipal level in Mexico," Doctoral dissertation, 2009.
[4] Grimm, N. and Shaw, N., "Using participatory action research to evaluate and improve change: Pilot implementation of electronic health records in rural Tanzania," *Medinfo 2007: Proceedings of the 12th World Congress on Health (Medical) Informatics; Building Sustainable Health Systems*, 2007.
[5] William, F. and Boren, S. A., "The role of electronic medical record in care delivery in developing countries," *International Journal of Information Management*, vol. 28, issue 6, pp. 503-507, 2008.
[6] Krishna, R., Kelleher, K. and Stahlberg, E., "Patient confidentiality in the research use of clinical medical databases," *American Journal of Public Health*, vol. 97, issue 4, pp. 654-658, 2007.
[7] Holden, R. J., "A theoretical model of health information technology usage behavior with implications for patient safety," *Behavior and Information Technology*, vol. 28, issue 1, pp. 21-38, 2009.
[8] Gillies, A. and Howard, J., "An international comparison of information in adverse events," *International Journal of Health Care Quality Assurance*, vol. 18, issue 4/5, pp. 343-352, 2005.
[9] Bruce, K. A., James, A. G., Stephen, R. L., George, L. R. and Lesley, M. R., "Challenges in health and health care for Australia," *MJA*, vol. 187 issue 9, pp. 485-489, 2007.
[10] Coiera, E., "Four rules for the re-invention of health care," *British Medical Journal*, vol. 328, pp. 1197-9, 2004.

[11] Sonia, F., Salvador, L. and Alicia, V., "Formal verification of websites," *Electron. Notes Theor. Comput. Sci*., vol. 200, pp. 103–118, 2008.
[12] Message, R. and Mycroft, A., "Controlling control flow in web applications," *Electron. Notes Theor. Comput. Sci.*, vol. 200, issue 3, pp. 119–131, 2008.
[13] Thomas, L. N., *Integrated Security Systems Design: Concepts, Specifications, and Implementation*, 1st edition. Butterworth-Heinemann, 2007.
[14] Ouma, S. and Herselman, M. E., "E-health in rural areas: Case of developing countries," *International Journal of Biological and Life Sciences*, vol. 4, issue 4, pp. 194-200, 2008.
[15] Jovanovic, N., Kruegel, C. and Kirda, E., "A static analysis tool for detecting Web application vulnerabilities," in *S&P'06: Proceedings of the 27th IEEE Symposium on Security & Privacy*, pp. 258-263, 2006.
[16] Wassermann, G. and Su, Z., "Static detection of cross-site scripting vulnerabilities," in *ICSE'08: ACM / IEEE 30th International Conference on Software Engineering*, pp. 171-180, 2008.
[17] Kruegel, C. and Vigna, G., "Anomaly detection of web-based attacks," in *CCS'03: Proceedings of the 10th ACM conference on Computer and Communications Security*, pp. 251-261, 2003.
[18] Ingham, K. L., Somayaji, A., Burge, J. and Forrest, S., "Learning dfa representations of http for protecting Web applications," *Computer Networks and Isdn Systems*, vol. 51, pp. 1239-1255, 2007.
[19] Chen, Y. and Malin, B., 'Detection of anomalous insiders in collaborative environments via relational analysis of access logs," in *CODASPY'11*, pp. 63–74, 2011.
[20] Li, X. and Xue, Y., "Protecting web-based patient portal for the security and privacy of electronic medical records," Department of Electrical Engineering & Computer Science, Vanderbilt University, 2012.
[21] Li, X. and Xue, Y., "Sentinel: Securing database from logic flaws in web applications," in *CODASPY '12*, pp. 25-36, 2012.