SECURITY OF THE SYSTEM FOR ELECTRONIC LEARNING

Igor Ristić[1*], Aleksandar Radonić[1]

[1]Faculty of Management, Sremski Karlovci, University UNION Nikola Tesla, Belgrade
e-mail: risticig@famns.edu.rs; aleksandar.radonjic@famns.edu.rs

Abstract: Security is one of great problems in the creation and implementation of e-learning solution. As an end user is not able to see these problems, they are often neglected. As all other applications, e-learning portals are widely distributed and available to a wide audience, which makes them vulnerable in a way. As the concept of Web applications is very spontaneously developed without a detailed plan, and with great effort invested in development of programming languages and interpreters for Web applications, there is a danger from manipulation by malicious users.

Keywords: *e-learning; education; security; attacks.*

I. INTRODUCTION

Basic security requirements are:

- Confidentiality – it refers to the belief that information and data, which imply something secret and private, will not be discovered by unauthorized persons, processes or devices. Namely, students need to be sure that their exams, which they have taken on-line, are kept private and that they can be accessed only by authorized users – teachers.
- Integrity – it refers to the belief that information and data will not be, accidentally or with malicious intent, modified or destroyed and that they will retain their accurate, correct and complete original form. Namely, the students need to be sure that their exams, taken on-line, reach the teachers in their original form.
- Availability – it refers to the belief that information and communication resources are available and reliable in a timely manner to authorized persons. Namely, the students need to be sure that they have a reliable and timely access to e-learning system when they want to take their exams in time.

Emergence of electronic communications, particularly the Internet, has significantly influenced the increase of frauds and identity thefts. For that reason, identity protection has become crucial in cyberspace. Basic and recognizable set of characters of an entity is what constitutes the identity, and it is also the thing that enables the others to be different from the rest. Such concept suggests that there aren't two same identities and that each identity is associated with a single set of characters that is unique to him. In on-line environment, the identities of users are digital identities.

The simplest way to ensure the protection of digital identities is the use of user login. All it takes is user ID and password of the user. As a result, user login provides three crucial access services of identity:

- Identification – recognizes the user as a true member of user association,
- Authentication – verifies user identity and
- Authorization – authorizes the access to specific resources.

Secure login system provides: control – review of user's on-line transactions; jurisdiction – linking user actions; recognition – elimination of activities that were rejected by the user

II. SECURITY MODEL OF DISTANCE LEARNING SYSTEM

A security model of distance learning system is based on role policy [4]. If it is assumed that U – is a set of users; G – a set of user groups, which consists of the following elements: G={administrator, teacher, student, guest}; P – is a set of authorizations to access the objects, presented in a matrix of access rights; S – set of user sessions in the system.

For these sets, the following relations can be defined [6]:

PA $\subseteq$ PxG - shows how the set of authorizations is reflected on the set of groups, where

*Corresponding author: risticig@famns.edu.rs

tools, indicated by those authorizations, are installed for each group; UA⊆UxG - shows how the set of users is reflected to groups, defining the tool to access the groups for each user.

Security rules of access policy control and define the following functions:

User: S →U – for each session, S function defines the user, who accomplishes this session with the system:

User(s)=u; group: S →P(G) – function defines tools from the set G for each session s, which can be simultaneously available to user in this session:

$$group(s) = \{g_i, |(user(s), g_i) \in UA\};$$

Authorization: S →P – this function will mark tools necessary for authorizations in sessions, authorization is defined as a set of authorizations of all groups, used in this session:

$$authorization(s) = U_{g \in group(s)} \{p_i | (p_i, g) \in PA\}$$

As a criterion of the security model, the following rule uses the following formula:

$$\forall u \in U, \exists p \in P, \exists s \in S$$
$$(u = user(s) \,\&\, p = operation_i(s)) \rightarrow$$
$$p \in authorization\ s(s)$$

Where the operationi(s) – is a function that will mark authorization for action number i.

In that way, the system is secure if any system users, who operates in session s, can accomplish actions, requiring the authorization p only in that event, and if p belongs to the set of available rights of session s.

III. APPLICATIONS FOR E-LEARNING COURSES

Many applications were developed as support in the preparation of *e-learning* courses. All those applications perform some common functions [5]:

Administration – application of administration is designed for the purpose of managing administrative information of an institution/organization. Administrative information is very sensitive, given that it refers to personal information. These tools are aimed at enabling the administrators to manage the important information of an institution/organization in an easy way.

Authorization of courses – many distance learning courses are designed to be accessible over the Internet. As a result of that, there is a need for rapid development of tools for multimedia courses.

Delivery of course content – after the *on-line* course is designed, it is necessary to deliver it by appropriate tools. These tools provide the students with the access to *on-line* courses over the Internet. On the other hand, students can have a variety of devices and tools and they can move them on various platforms.

Synchronous communication – some tools are made in such a way to support synchronized activities between instructors and students. Typical example of that is a tool for a video conference that is designed in such a way that activities, such as visual communication "*face-to-face*" and audio communication, require synchronization between communication parties. Although these tools are useful, they also have their disadvantages. First of all, they require the expansion of network scope, which will support communication requirements.

Multimedia lectures – some applications offer tools for synchronization of video "*slide show*" presentation. These multimedia applications aim at providing the learning environment that is similar to the traditional learning environment.

Assessment of students' knowledge – it is a challenge for each instructor to assess the knowledge of students and determine how well the student handles *on-line* courses. Different tools that support *on-line* testing were made. These tools tend to focus on the creation of *on-line* exams or to monitor students' logins.

IV. TYPES OF ATTACKS ON E-LEARNING SYSTEMS

Some of the types of attacks are browser kidnapping, theft of cookies, theft of database, changes of access rights, i.e. DoS (Denial of Service), SQL/HTML injection, cross-site inclusion etc. [3] HTML injection is an attack that is done by entering HTML/JavaScript code in the text fields that should, later on, be shown on one of the pages of Web application [2].

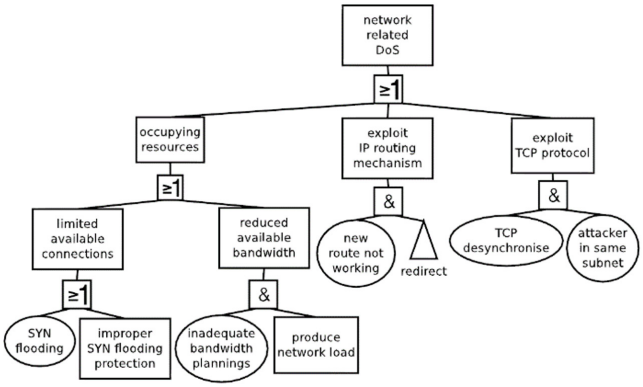

Figure1. Fault tree excerpt for network related denial of service (DoS)

Such failures are often used for so-called "session thefts". Each user who visits the page that contains a malicious code, including the administrator, is a potential victim of the attack. The attacker can write a code by which he can record all cookie variables from user's browser on a remote location, including those that contain sensitive data, such as session's identification number. The attacker can then present those cookie applications as his own, and the application will, by the identification number of attacker's session, see the attacker as the one whose cookie is stolen. In that way, if the attacker steals administrator's cookie, he will have all the privileges in the application including editing and deleting the content and user accounts. SQL injection is another very frequently used technique [1]. It is relatively similar to HTML injection, but it is even more dangerous. This attack allows the attacker to edit or read arbitrary entries from SQL base, and this can be used in many ways, such as by creating new administrator's user account.

Security gaps are found in two basic ways:

a) By brute-force access, i.e. by testing all the possible failures "blindly".

b) By reading the application code and analyzing it.

From all this, it follows that only well-checked cryptographic algorithms should be used (RSA; DES, Blowish, MD5, SHA,…), instead of writing own solutions. Verification of important information should be done on server's side that is opposed to client-side of language. Thus, security-critical information must not be verified, and an important code must not be located in languages that are performed on client-side (JavaScript, Java, VBScript, Flash, …). All input data should be "cleaned" from potentially dangerous parts by deleting HTML and JavaScript from the text that the user sends to be presented on web pages or provides the translation of special characters into equivalents for the use in SQL queries. So, web applications are most frequently written in languages whose code is never compiled, but it is interpreted, i.e. their code is readable. One of the most common techniques for hiding the code is obfuscation. Obfuscation is coding (not encryption) of the code, for the purpose of reducing its readability. It is aimed at reducing "the sense" of the written code by changing the names of variables, functions, classes, to replace as many codes by equivalent etc. In case of using PHP, most frequently used solution is ZendGuard that partially complies the code into so-called bytecode, in addition to obfuscation. It should be stressed that regardless of all this, there is no completely secure way to hide a code if the executable file is in question.

V. SETTING THE SECURITY ASPECTS OF APPLICATIONS FOR E-LEARNING COURSES

Web sites consist of client and server components. By this classification, it can be said that the system is completely secure as long as it is in the context of the client, because it actually does not have anything to do with the server. However, in the moment when a user-defined process takes place on the server, one site becomes vulnerable.

What happens in the process of broadcasting one static HTML page? Client application sends a request to *web server* and web server responds by finding and broadcasting the requested *HTML* document. In such process, there is no room for anything, except for the mentioned series of activities and, because of that, this application is secure.

In process of creating and broadcasting dynamic web side, this process has a few more steps. In the beginning, and here, the client application requires a particular document, but the server, instead of finding and forwarding that document, forwards the entire request to server script and it processes it and transmits on the output (to the client). This processing is a key point for security of one *web* application, because if the user succeeds to infiltrate its part of the code in server script, it will have "unlimited" possibilities to manipulate the server.

Therefore, the input is the most vulnerable part of the application. For that reason, it is the most important to be sure about everything that enters the application; and that security will, of course, be accomplished by controlling all "inputs".

A. What are the inputs into a web application?

In order for the user to reach the server code of an application (through that application), it is necessary to turn to it through some parameters. These parameters, usually, reach the application through the forms (*post*) or parameterized *URL string (get)*. When some of these parameters reach the server, server puts it into an appropriate variable. These variables are unique and available to the complete context of application and, for that reason, they are called superglobal variables.

Most of the superglobals, in case of each request and response, pass the way from a client to server and vice versa and, therefore, they are considered unclean and they need a special treatment in order for their use to be secure.

B. The concept of black and white list

A lot of intrusions into application occur through controls that are, generally, an insecure source. For that reason, in case of every such information input, a certain filtration is performed. Primarily, it should be devised what is to be filtrated.

When you are filtrating data, the application can be said to do one of the two things:

- not let anyone in who does not meet particular conditions,
- let only those who have met particular conditions.

These two concepts are called black and white list.

Difference between these concepts is in the fact that black list requires much less attention, because, after we list the objects that do not have the access to the structure, it will be available to all other objects even if the list is not timely refreshed.

On the other hand, white list requires a more regular refreshing, depending of the frequency of objects. White list is considered to be a better security concept than black list, because the input is limited only to the values expected, and thus the undesired object has much less chances to pass.

C. Input

The first vulnerable point in the system is superglobal variable, and that is simultaneously the place where server code has a possibility of a control.

The first thing that can be controlled is whether the user is appropriate or not. In case that it is a user that is not registered in the system, it can be verified from where it came. The location from which the user came is called **referrer**.

If it is expected for the user to be registered in the system, the systemic verification is done (through the *cookie, session* or base).

When the user is verified, the following point is the input itself, i.e. superglobals that contain that input. In *web* applications, the user can cause serious harm only through a server or

SQL script. For that reason, such an input is most frequently necessary to be prevented; the best way is to forcing the user to enter exclusively valid content.

VI. CONCLUSION

Development of many e-learning courses is most frequently based on technical details, as well as the manner of their delivery. Security, as the need of these systems' existence is frequently neglected. Namely, the role of security in e-learning systems is to provide a secure „*end to end*" session between students and *e-learning* network, where security is treated as a technical element.

Observed from the perspective of students, security in *e-learning* environment is focused on something else. The ability of a student to manage his own space, especially when personal information is shared, is very important. In *e-learning* environments, when physical interaction practically does not exist, the confidence is essentially important.

REFERENCES

[1] Anley, C.: Advanced SQL Injection In SQL Server Applications. Tech. rep., NGSSoftware Insight Security Research (NISR), 2021.
[2] Borcea, K.; Donker, H.; Franz, E.; Ptzmann, A.; Wahrig, H.: Towards Privacy-Aware eLearning, PET 2005, Springer, 2006, no. 3856 in LNCS, pp. 167-178.
[3] Jeremiah Grossman & Lex Arquette, Defcon 9 Web Aplication Security "In theory & practice", www.whitehatsec.com
[4] Kajava, J.: Security in e-Learning: the Whys and Wherefores. In: European IntensiveProgramme on Information and Communication Technologies Security (IPICS'2003),4th Winter School, 2003
[5] Learning Circuits [Online], "Field Guide to Learning Management Systems", 2021, Retrieved November 25 2022, URL: http://www.learningcircuits.org/
[6] OWASP Top Ten Project [Online Report], Open Web Application Security Project - the open application security community, 2021, Retrieved November 25 2022, URL: https://owasp.org/Top10/