УДК 004.9/004.6:004

S.V. Olszewski, D.I. Tarnavskyi, Y.S. Ivzhenko

WEB-APPLICATIONS PROTECTION ESTABLISHMENT

Annotation. *The modern world carries thousands of threats and potential dangers literally at every step and every moment of time. The World Wide Web, which has become an integral part of our lives, is no exception.*

Keywords*: information protection, SQL injection, cross-directory attack.*

Introduction

Insecure software undermines our finances, health, defense, energy and other important infrastructures. As software becomes more and more complex and related, the complexity of achieving application security grows exponentially. The rapid pace of modern software development processes makes the most common risks needed to identify and resolve, with what's fast and accurate. [1]

Google reported the safety of websites in 2016, announcing an increase in the number of broken sites by 32% in 2016 compared to 2015. Also, the trend is not expected to decline soon. [2]

The OWASP (Open Web Application Security Project) community is involved in the classification of Web application attacks and vulnerabilities. This is an international nonprofit organization focused on analyzing and improving software.

OWASP has created a list of the 10 most dangerous attack vectors on the web application, the list being called OWASP TOP-10, and it focuses on the most dangerous vulnerabilities. This project, first published in 2003 and regularly updated. It aims to increase awareness of application security by identifying some of the most critical risks to organizations

Technique of illegal extraction of confidential information and protection mathods

SQL injection is one of the most common methods of hacking applications running on databases, based on the implementation of an arbitrary SQL code request. Thus, changing the input parameters by adding in them the structures of the SQL language causes a change in the logic of executing the SQL query (in this example, instead of the news with the given identifier, all available news in the database will be

selected, since the expression 1 = 1 is always true).

A Simple SQL Injection Example:

*String stmt = "SELECT * FROM users " +"WHERE username='" + username + "' " + "AND password='" + password + "';";.*

Intruder input:

"jakob" + "' OR 1=1;-- "WHERE username='jakob' AND password='' OR 1=1;-- ';

To protect against SQL injections use:

- shielding single and double quotes;
- checking of type or data type typization;
- checking of the structure of input data using regular expressions.

Cross Site Scripting is a type of vulnerability in interactive information systems. XSS occurs when the page that was generated by the server for some reason fall into user scripts. The specific of these attacks is that instead of direct attacking the server, malicious users use a vulnerable server to attack the user.

As browsers do not allow interaction between sites (cross-site access), intruders use different techniques to implement cross-site scripting:

1) Saved intersite scripting matches the entry of a malicious script by a web application to steal temporary files that contain the session ID. Such a script redirects the web application to a web page hidden from the user, the URL request will contain temporary victim browser files as a query parameter. Once an attacker receives a temporary cookie URL, unauthorized access to the user session may occur.

2) Reflected cross-site scripting matches the reflection of an infected script from a web server, such as an error message, a search result, or any other response that includes a portion or all of the data sent to the server as part of the request. Reflected attacks are delivered to the victim (user) in another way, in the form of an email or as a link to another website. When a user clicks on such a malicious link, fills in a special form on the website, or even simply browses the malicious web resource, the affected code gets to the vulnerable website, which reflects the attack back to the user's browser. Then, the victim's browser executes this code because it came from a "trusted" server. Reflected cross-site scripting is also known as unstable.

3) Cross-site scripting based on an object model of a document (OML)

is a type of attack, when attacking actions are performed as a result of modifying the ODM environment in a victim's browser using a client-side script, that is, in this case, the user code executed in an unexpected manner . In this case, the web page remains unchanged, but the code stored on the victim page is executed according to the malicious modification of the OMD environment.

Example: Submit the form using invalid data. It is redirecting to http://acme.com/login.php?msg=Invalid%20login%20data.

The msg parameter is enabled in HTML. The attacker uses the msg parameter to add a malicious script:

```
window.onload = function( )
{
  document.forms[0].action= 'http://evil.com/steal_data.php';
};
```

http://acme.com/login.php?msg=%3Cscript%3Ewindow.onload%20%3D%20function%28%29%20%7Bdocument.forms%5B0%5D.action%3D%27http%3A%2f%2fevil.com%2fsteal_data.php%27%3B%7D%3B%3C%2fscript%3E

To protect use:

- encryption of each variable that is part of the HTML;
- checking the user's input, creating whitelists.

The **Counter Site Request Forgery (CSRF** also known as **XSRF)** is a kind of attack on website visitors that uses the disadvantages of the HTTP protocol. If a victim enters a site created by an attacker, a person is secretly sent a request to another server (for example, to a payment system server) that performs some harmful operation (for example, money transfer to the account of the attacker).

Most **CSRF** prevention methods work by embedding additional authentication data in queries that allows the web application to detect requests from unauthorized locations.

Cross-directory attack. An attacker's access to files and folders stored outside the root directory. Possible manipulation of variables that refer to files using "dot-dot-slash" (../) sequences. Web application that shows the user's home directory files:

http://acme.com/list_user_files.php test.txt

hello_world.txt http://acme.com/show_file.php?file=test.txt

The vulnerability lies in this case:

http://acme.com/show_file.php?file=../../../../etc/passwd

To protect using: "whitelisted" user input, identifying files for temporary access.

Incorrect session management includes cryptographically weak session identifiers (Session ID's), Sotsitelnaya engineering, etc. Possible methods of prevention:

- attaching the session identifier to the IP address;
- cryptographically persistent session identifiers (using proven encryption algorithms).

Conclusions

Web resource security is a process that combines a certain set of actions. The developed system is first investigated for safety purposes, then a series of measures and works being done to achieve it are determined.

REFERENCES

1. https://www.drasecurity.com/secure-design-review.html.
2. https://webmasters.googleblog.com/2017/03/nohacked-year-in-review.html.
3. https://uk.wikipedia.org/wiki/OWASP.
4. M. Vieira, N. Antunes, H. Madeira Using web security scanners to detect vulnerabilities in web services// IEEE/IFIP International Conference on Dependable Systems & Networks (2009).
5. N. Antunes, M. Vieira, S.Q.L. Detecting injection vulnerabilities in web services// Fourth Latin-American Symposium on Dependable Computing (2009).

 ISSN 1562-9945