VOL. 10, NO. 18, OCTOBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

SECURITY HOLES IN MANUSCRIPT MANAGEMENT SYSTEMS

Dmitry Sergeevich Silnov
Department of Information Systems and Technologies, National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Moscow, Russia
E-Mail: ds@silnov.pro

ABSTRACT

Today’s web-oriented programming languages are simple in nature, thus increasingly used in developing websites. However, simplicity has also weakened data security in the content management systems of these sites.

Keywords: SQL-injection, software bugs, PHP errors.

INTRODUCTION

Errors are frequently found in web-oriented software. This is directly related to the skills of those that develop such software. It is wrong to think that errors are peculiar only to web-engines designed by amateur or individual programmers. Attackers exploit source code errors to carry out SQL injection attacks and gain access to confidential data. Such errors are committed both by professional PHP programmers and beginning developers. Evidence to this is the fact that such errors are regularly detected both in professionally developed software systems [1] [2], and in software designed by amateur or individual programmers. Let us examine closer errors in software used by various journals to automate manuscript management.

PUBLISHING SOFTWARE: ANALYSIS

The most efficient way of detecting bugs in a software system is to analyse the source code. However, source code analysis is not always possible especially when there is no open access. Moreover, a very large source code could hinder its quality analysis: when analysing such large code, the computer security specialist may miss important parts of the code, thereby not detecting bugs in the source code.

Analysis performed without source code access can be both automated (when special software generates requests to the website and searches through possible parameters in order to detect errors) and manual to detect vulnerabilities in the web engine code.

The author of this article manually analysed the web engine of the site jatit.org: the official website of the Journal of Theoretical and Applied Information Technology. It probably has its own individually-developed web engine. At the same time, many journals use the same type of software Open Journal Systems (OJS) [3]. On one hand, OJS is convenient to use, as it incorporates modern publication requirements when handling manuscripts. On the other, this is a finished product that is difficult to be modified. Analysis was performed using the standard tools of Firefox browser with which transmission of parameters under POST and GET requests was monitored.

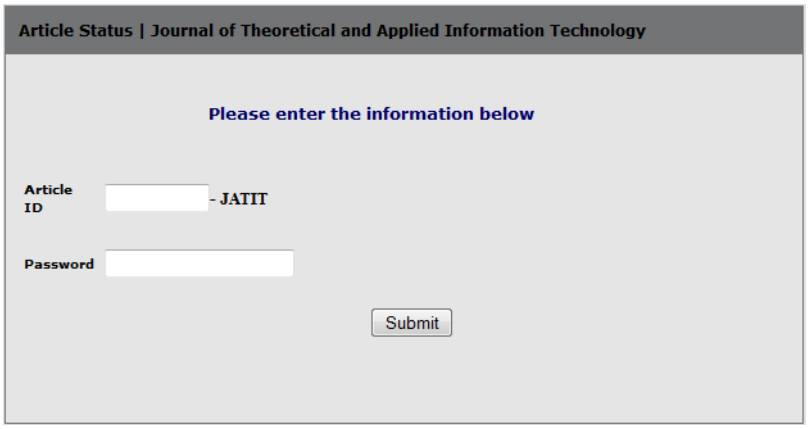
Article Status | Journal of Theoretical and Applied Information Technology
Please enter the information below
Article ID
- JATIT
Password
Submit

Figure-1. Login page to check manuscript status.

METHODS

The page for checking the status of manuscripts enables authors to monitor the status of their papers. The status can be “Under Blind Review Process”, “Accepted for publication after Peer double blind Review”, “Conditionally Accepted for publication after Peer double blind Review” or “refused”. This is not the full list. In order for authors to check the status of their manuscripts, they need to know its ID and password (Figure-1). These data are generated randomly and sent to the corresponding author by e-mail.

It would seem that without a password, an author will not be able to check the status of his article, not to talk of that of other authors. Actually, the web engine is designed in such a way that you can check the status without entering a password. This is so because validation of article ID and password is done in one PHP file, while the status is displayed in another file (Figure-2).

This website does not use sessions, which would allow user authentication in the script process_manuscript_status.php, and checks the relevance of this authentication in manuscript_status.php before displaying the status details of the requested article.

A specially formed request to manuscript_status.php will retrieve information on any manuscript in the system. We will use programming language PHP with CURL module, with which one can create requests to remote web pages, imitating a web browser. The source code looks as follows:

```
$article_id="123"; // Article number
$url = 'http://www.jatit.org/manuscript_status.php';
$useragent="Mozilla/5.0 (Windows; U; Windows NT 5.1; ru; rv: 1.9.2.2) Gecko/20100316 Firefox/3.6.2 (.NET CLR 3.5.30729)";
$ch = curl_init();
curl_setopt($ch, CURLOPT_COOKIESESSION, TRUE);
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_COOKIEFILE, "./cookiefile");
curl_setopt($ch, CURLOPT_COOKIEJAR, "./cookiefile");
curl_setopt($ch, CURLOPT_USERAGENT, $useragent);
curl_setopt($ch, CURLOPT_COOKIE, session_name() . '=' . session_id());
curl_setopt($ch, CURLOPT_FOLLOWLOCATION, 1);
curl_setopt($ch, CURLOPT_RETURNTRANSFER,1);
curl_setopt($ch, CURLOPT_INTERFACE, $local_ip);
curl_setopt($ch, CURLOPT_URL,$url);
curl_setopt($ch, CURLOPT_POSTFIELDS, "info=".$article_id."&id=&email");
```

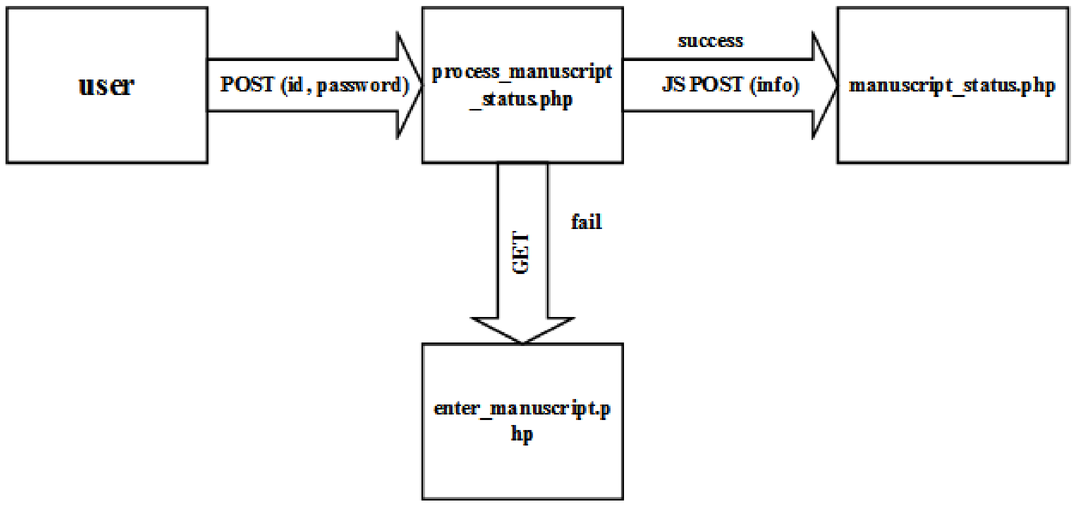

Figure-2. Request workflow.

```
$out=curl_exec ($ch);
print $out;
```

A more detailed analysis shows that the script manuscript_status.php is also vulnerable to SQL injection [4]. This allows an attacker to modify an SQL request to the target system with the aim of retrieving the database contents. However, the built-in security system filters separate words such as UNION, and some others in POST variables. This prevents SQL injection.

The problem of SQL injection is also peculiar to the script proc_paper.php, which is called by POST request when submitting a manuscript for review. For example, in the description field, you may place a text containing special characters, such as “’” (quote), which would allow you to carry out an SQL injection attack.

These problems are addressed via either special server configurations, which have been partly done, to filter separate words, such as "UNION" in variables, or by using PHP configurations, which can forcibly convert special characters to HTML entities. If such features are not present, then special characters can be added by a PHP programmer by using the function htmlspecialchars [4].

ANALYSIS OF RESULTS

As shown by surface analysis, such web engine is used by more than one site. For example, the Journal of Engineering and Applied Sciences (http://www.arpnjournals.com/jeas/) uses a similar system, but without separate elements that are most vulnerable to the problems mentioned. The feature for checking the status of manuscripts is not active. The file manuscript_status.php is inaccessible to users. Therefore, this journal employs a more serious security approach.

The use of both a standard software system and a non-professional software is equally interconnected with safe programming. Lack of attention to security issues in such systems may enable an advanced author to exploit security holes to update the status of his/her manuscript and publish it, bypassing peer review.

REFRENCES

[1] Open Journal Systems Multiple HTML Injection Vulnerabilities. Retrieved June 18 2015 from http://www.securityfocus.com/bid/43054/info.

[2] Multiple vulnerabilities in Open Journal Systems (OJS). Retrieved June 12 2015, from https://www.htbridge.com/advisory/HTB23079.

[3] Open Journal Systems. Retrieved June 12, 2015, from https://pkp.sfu.ca/ojs/.

[4] Htmlspecialchars.Convert special characters to HTML entities. Retrieved June 7, 2015, from http://php.net/htmlspecialchars.