International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 4 Issue 4, June 2020 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Data Security: String Manipulation by Random Value in Hypertext Preprocessor

Danial Kafi Ahmad[1,2], Zul Hilmi Abdullah[1], Siti Nuraini Ahmad[3]

[1]Faculty of Information Technology, INTI International University, Nilai, Malaysia
[2]Language Centre, National Defence University of Malaysia, Kuala Lumpur, Malaysia
[3]Putra Business School, Universiti Putra Malaysia, Malaysia

ABSTRACT

Hypertext Preprocessor (PHP) and Hypertext Markup Language (HTML) were important as scripting languages in most of the web based development. As an open source type, it has benefited educators and web developers in either education or commercial context due to their easy accessibility. However, there were many concepts and mechanisms that could be learnt and explored in order to produce quality system design in this respective language. As web based system transmit and exchange data within a vast network of Commercial Interconnected Network (Internet), the data were exposed to many attackers who wish to steal the data, therefore the security aspect which focusing on protecting the data technically (automated computation) should be taken into account when designing the system, apart from the policies, rules or laws enforcement in cyber security environment. In this experiment, a light data manipulation technique were developed to convert the string (user input) into different forms of text representation of numerical value.

KEYWORDS: *Iteration, Web, Data Manipulation, Random*

How to cite this paper: Danial Kafi Ahmad | Zul Hilmi Abdullah | Siti Nuraini Ahmad "Data Security: String Manipulation by Random Value in Hypertext Preprocessor" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-4 | Issue-4, June 2020, pp.873-878, URL: www.ijtsrd.com/papers/ijtsrd31283.pdf

IJTSRD31283

I. INTRODUCTION

Web based system had become popular among people globally especially the E-Commerce type. This could be seen via the increasing numbers of E-Commerce users globally whereby they perform the purchasing and selling transactions online and this had shown the importance of the system and it were expected to be more demanded by the users in future [1]. On a different note, several E-Commerce would have different designs and perhaps different quality as perceived by the users. However, the main concern among all of the users is the security aspects (either technical or legal context) of the system that they used [2]. For example, the user would have to give their sensitive information such as login id, password and payment information etc., and this would be every user's concern to protect their data, and this could be achieved by deploying security mechanism (technical computation) such as data manipulation in within the system. The technique and design of data manipulation could be varied, and their variety could be important as to create more options of security mechanism that would be deployed in the system. Moreover, it could be used by educators in the teaching and learning processes in order to deliver the concept to the students in security topic. In addition to that, techniques of data manipulation which deployed by several scripting and programming languages should be studied specifically at large, regardless of their efficiencies as different languages could interpreted the techniques in different manner in terms of semantic or syntax [3]. The limitations of the variety techniques could be an opportunity for the researcher, educator or developer to understand deeply regards to the techniques developed whereby they could study the limitations and came out with the solution. It is believed that a good design of security technique (technical) would derived from a good understanding of the technique, and the understanding could be achieved via thorough or details study of variety techniques in terms of their mechanism[4][5]. Therefore, it is a good action to investigate and develop more techniques or theories consistently and rapidly of data manipulation in security as to gain more vast understanding of the concept [6]. In this research experiment, a light module of data manipulation were developed using an open source scripting languages which are Hypertext Preprocessor (PHP) and Hypertext Markup Language (HTML) in order to see the ability of the developed technique and algorithm to manipulate the input data by the user in technical computation context. The languages were deployed due to their easy accessibility, functionality and stability.

II. Methodology

SDLC

Software Development Lifecycle (SDLC) were implemented in this research development as to achieve systematic

approach [7]. A study said that systematic approach is important in a context of Software Engineering as it could lead to better quality software produced. However, in spite of the systematic approach, a non-plan driven were deployed in the development. This means that the ad hoc approach were used in the development whereby the executed stages or phases were determined by the requirement condition during development. In this research experiment, an incremental model which comprise of Specification, Development and Validation phases were implemented as a software process model as the development would provide more flexibilities and agility during development [7].

Software Process Model (Incremental)
Specification
Development Tools

Figure1 shows the tools or platform that were used in this research experiment:

- Apache of XAMPP Package (Server)
- Notepad Application
- Browser (Client)
- Local Disk C
- XAMPP Control Panel

Figure1. Tools for development.

Figure2 shows the requirement of the developed program:

- The module should be able to accept input keyed in by the end user.
- The module should be able to convert the initial input into different form of expression of values.
- The module should be able to generate random value via rand() built in function.
- The module should be able to redirect to main page after data manipulation computation
- The module should be able to display the manipulated input as a string type
- The program is able perform a single complete cycle in within an expected duration.

Figure2. Requirement for the program module.

Architecture-Client Server

In this research experiment, the program module would work in within the client server architecture. Figure3 shows the component of the architecture. The client would request the web page form the server, and the server would response to the client via the requested web page with the condition of the existence of the web page [8].

Client ⟷ **Server**

Figure3. Client Server Architecture.

The architecture are well known as it were deployed in any web based system. The exchange of web pages among the components reflects the concept of the architecture. However, the program computation (PHP) were written in the server side making it as a server side scripting language.

Pseudocode

Figure4 shows the pseudocode of the module which focusses on the program computation (PHP). It visualize the general flow of the system and does not imply to the executable means. This means that a semantic and some syntax issue could be trace via the pseudocode instead of runtime error. This also had reflects the use of pseudocode in supporting testing as a tool.

```
BEGIN
        Read input from user
        Loop start
        for($i=0;$i<strlen($input);$i++)
        Generate random number from 0 to 9
        Assign the random number to input with
        offset ($i value)
        Loop end
        Display the manipulated input (value)
        Redirect to form page
END
```

Figure4. Pseudocode of manipulation page (encnew.php)

Development
Code

The module were developed based on several functions or statements of PHP in required sequence or order as according to the program requirements. Table 1 shows the functions or statements that were deployed in the developed program.

Function/Statement	Description
<?php ?>	Indicating the start and end point of the PHP program
echo()	Display output
strlen()	Return the number of counts of string
$input	A variable which hold the string data type keyed in by the user as well as overwritten string after manipulation
$_POST[]	Retrieve the data form the textbox input of HTML type
rand()	Return a random number (integer) in within a certain range
$input[]	Defining and overwriting character at specific index (offset)
header()	Redirect to required page
<html></html>	Indicating the start and end point of HTML program
<form></form>	Indicating the start and end point of form of HTML
method="post"	Retrieve data
action=""	Redirect to other webpage
name=""	Define name for the input
value=""	Define text on the input type (*submit type)
<input type="text">	A graphical textbox to accept input
<input type="submit">	A graphical button to transmit data

Table1. Purpose of program components.

Installation and Configuration

Configuration would be required in a case of unavailable port, and if yes, then the port would need to be changed to available port. "Typical" installation of the developed module would not be required as the program were typically written in the server folder, and the user would be able to view the

program via the web browser. In this research experiment, the localhost were deployed instead. Figure5 shows the common component that involved in the configuration:

- XAMPP Control Panel
- *httpd.conf* (Apache)
- Services (XAMPP Control Panel)

Figure5. Configuration components

In this experiment, the *Listen 80* were changed to *Listen 8080* in order to provide available port. The configuration would be depending on the machine whereby different machine would may require no changes of the *Listen*. The configuration occur in the *httpd.conf* file.

Validation

The requirements as listed in Figure2 were tested against the testing activities of test level, and test technique. These were required in order to make sure the program achieved the required Verification & Validation (V&V) [9]. The testing is mainly focus on the functional requirements in order to make sure the computation would be able to work as expected.

Test Level

Figure6 shows the test level model which were carried out in the research experiment [9].

- **Component test**
  (see section 3.2) verifies whether each software →component correctly fulfills its specification.
- **→Integration test**
  (see section 3.3) checks if groups of components interact in the way that is specified by the technical system design.
- **System test**
  (see section 3.4) verifies whether the system as a whole meets the specified requirements.
- **→Acceptance test**
  (see section 3.5) checks if the system meets the customer requirements, as specified in the contract and/or if the system meets user needs and expectations.

Figure6. The test levels.

Test case were used in this research experiment in order to show the executed testing of the requirements with their condition/information such as Test Case ID, Type, Objective, Input, Procedure, Expected Output, Actual Output and Result. These information were important in order to deliver the relevant points of the test case [9].

Test Case ID	The unique ID
Type	The test type (Functional or Non-Functional)
Objective	The purpose of testing
Input	The input (String type etc.)
Procedure/Steps	List of actions required
Expected Output	The expected displayed results prior to execution
Actual Output	The displayed results of execution
Result	Test case results

Figure7. Test Case template.

III. DISCUSSION

Algorithm

Coding

```
<html>
<form action="encnew.php" method="post">
<input type="text" name="input">
<input type="submit" value="manipulate">
</form>
</html>
```

Figure8. Form Page (encnewinput.html)

Figure8 shows the code or scripts of form page (encnewinput.html) which were written using HTML scripting language. The language were written in order to create a form of Graphical User Interface (GUI) which enable the user to insert data to be computed in within the system. The input type of "text" and "submit" were deployed to accept input and transmit the data (input by user) to other web page respectively. The transmission of data was done via a method post and action in within the HTML tag,

```
<?php
$input=$_POST['input'];
for($i=0;$i<strlen($input);$i++){
$input[$i]=rand(0,9);}
echo $input;
header("refresh:1.5;url=encnewinput.html");
?>
```

Figure9. Manipulation page (encnew.php)

Figure9 shows the code of the manipulation page (encnew.php). The program is important as the main computation or functional (string manipulation) of the developed program is in within the program. Figure10 until Figure14 shows the explanation of the codes in within the program.

Code 1:

$input=$_POST ['input'];

Description 1:

The scripts would take an input from the form text type of key name "input" using the method POST. The input then were assigned to the variable $input

Figure10. Code 1 description.

Code 2:

for($i=0;$i<strlen($input);$i++)

Description 2:

for() looping were used I order to compute the data or string manipulation in few iterations, whereby the iteration would be determined by the number of characters (input by user) including white space. For example if the input would be ***abc***, then the number of iterations would be 3 due to the length of the input is 3. However if the input were ***ab c***, then the iteration would be 4 due to the characters count is 4. The number of characters were computed using the function of strlen($input).

Figure11. Code 2 description.

Code 3:

$input[$i]=rand(0,9);

Description 3:

Assignment operators were implemented in this line (within the statement execution of the loop), whereby it consists of

Left Hand Side (LHS) and Right Hand Side (RHS). The statement would be executed if the loop return the TRUE value. In this line, the RHS would be executed followed by the LHS. This means that a random value from 0 to 9 would be generated and were assigned to the specific character (offset) of the string (input by the user). For example, the input is ***abc*** then the key or index for the input would be 0→a, 1→b and 2→ c. If the generated value would be 8 and it was in the first iteration with the $i is 0, then abc would become **8bc**. On the second iteration, the $i would be 1 and assuming the generated value would be 5, then the 8bc would become **85c**, and this followed by the last iteration with value of $i of 2, and assumed the generated value would be 0, then 85c would become **850** followed by the loop termination. The values (**8bc, 85c, 850**) were stored in the memory in sequence (overwritten until the last iteration, in this case, **850** would be the final value). Figure13 shows the illustration of the iteration which has been described.

Figure12. Code 3 description.

**Assuming following condition in the first run, second run and third run:*

1st run

- rand(0,9) produce 8.
- $input hold *abc*

```
$input[$i]=rand(0,9);
    $input[0]=8;
```

2nd run

- rand(0,9) produce 5.
- $input hold *8bc*

```
$input[$i]=rand(0,9);
    $input[0]=5;
```

3rd run

- rand(0,9) produce 0.
- $input hold *85c*

```
$input[$i]=rand(0,9);
    $input[0]=0;
```

****$input now hold 850***

Figure13. Elaboration of array redefinition of specific offset in within the loop.

Code 4:

```
echo $input;
header("refresh:1.5;url=encnewinput.html");
```

Description 4:

In this line, $input would hold the final value (in this case it would be **850,** as the final value on ***Description 3***) and were displayed as an output for 1.5 seconds. This followed by the redirecting to form page (encnewinput.html). However, if there is no input by the user, the output would be blank as the loop condition is not met and the $input is holding null value.

Figure14. Code 4 description.

Graphical User Interface

Below figures shows the output of all the versions of the program.

Figure15. Output of form page.

Figure16. Output of manipulate page with (abc) as an input.

850

Figure17. Output sample of a complete run with input (abc).

Testing (Validation)

The testing were done via static technique and dynamic technique. In this research experiment, the pseudocode and the code were used as a tool in static testing, whilst the execution of code and their computed result were used in dynamic testing. Both were required in order to complete the testing processes aa suggested by the experts [10]. The testing were documented in form of test cases as shown in table 4 until table 8.

Unit Testing

Test Case ID	**F-1**
Type	Functional
Objective	To test the functionality of text field of encnewinput.html program
Input	String Type Data (abc)
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	The input of (abc) would change to 850 and were displayed.
Actual Output	850
Result	PASS

Figure18. F1 Test case.

Test Case ID	**F-2**
Type	Functional
Objective	To test the functionality of the random number generator function and the string (input) displayed
Input	String Type Data (abc)
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	The input of (abc) would change to 851 and were displayed *The random number generated on other run might be different.
Actual Output	851
Result	PASS

Figure19. F2 Test case.

Test Case ID	**F-3**
Type	Functional
Objective	To test the functionality of the redirecting function
Input	String Type Data (abc)
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	The input of (abc) would change to 551 and were displayed, then form page appeared after 1.5 seconds. *The random number generated on other run might be different.
Actual Output	851
Result	PASS

Figure20. F3 Test case.

Test Case ID	F-4
Type	Functional
Objective	To test the functionality of loop iteration
Input	String Type Data (abcdef)
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	The input of (abcdef) would change to 386372 and were displayed. *The random number generated on other run might be different and the number generated is 6 (count).
Actual Output	386372
Result	PASS

Figure21. F4 Test case.

Test Case ID	NF-1
Type	Non-Functional
Objective	To test the response time of text field displayed of encnewinput.html program
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	Keyed in data appeared instantly in the textbox
Actual Output	Keyed in data appeared instantly in the textbox
Result	PASS

Figure22. NF1 Test case.

Test Case ID	NF-2
Type	Non-Functional
Objective	To test the processing time of functionality of the random number generator function and the string (input) displayed
Input	String Type Data (similar as to previous run)
Procedure/Steps	1. Enter data 2. Click submit button
Expected Output	851 displayed in within the timeline.
Actual Output	851
Result	PASS

Figure23. NF2 Test case.

Test Case ID	NF-3
Type	Non-Functional
Objective	To test the processing time of redirecting function
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	551 displayed in within the expected timeline.
Actual Output	551
Result	PASS

Figure24. NF3 Test case.

Test Case ID	NF-4
Type	Non-Functional
Objective	To test the processing time of loop iteration
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	386372 displayed in within the expected timeline.
Actual Output	386372
Result	PASS

Figure25. NF4 Test case.

Integration and System Testing

Test Case ID	F-1-ISY
Type	Functional
Objective	To test the linking in between encnewinput.html and encnew.php
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	encnew.php page displayed along with manipulated data (356)
Actual Output	356
Result	PASS

Figure26. F1-ISY Test case.

Test Case ID	F-2-ISY
Type	Functional
Objective	To test the linking in between encnewinput.html and encnew.php via header()
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button 3. Wait for 1.5 seconds after encnew.php page
Expected Output	encnewinput.html displayed
Actual Output	encnewinput.html displayed
Result	PASS

Figure27. F2-ISY Test case.

Test Case ID	F-3-ISY
Type	Non-Functional
Objective	To test the linking processing time in between encnewinput.html and encnew.php
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click manipulate button
Expected Output	encnew.php page displayed along with manipulated data (356) in within expected timeline.
Actual Output	356
Result	PASS

Figure28. F3-ISY Test case.

Test Case ID	F-4-ISY
Type	Non-Functional
Objective	To test the linking processing time in between encnewinput.html and encnew.php via header()
Input	String Type Data
Procedure/Steps	1. Enter data 2. Click submit button 3. Wait for 1.5 second after enc3.php page
Expected Output	inputenc.html displayed in within a timeline
Actual Output	nnputenc.html displayed in within a timeline
Result	PASS

Figure29. F4-ISY Test case.

IV. FUTURE ENHANCEMENT

The developed module could benefited educators, developers and other stakeholders in variety ways. However, it would require enhancement and extension especially if implemented in commercial context. For example, the manipulation could be enhanced by embedding special and additional characters which is greater than the actual count of the string, as well as multiple layer of manipulations. In addition, improvement in terms of generated number repetition could be improved and enhanced. As the developed technique is rather to provide the important finding on basic string manipulation concept and tool in context of PHP rather than being treated as a readymade tool for commercial, therefore this module could be treated as a foundation towards a good design of data manipulation in complex system.

V. CONCLUSION AND RECOMMENDATION

In this research experiment, it was found that the PHP language were able to compute the required data manipulation as according to the developed technique, by deploying the function of random number generator, in within a decent timeframe and output, and this has been the important finding in computer security in technical context. As most of the system would require a security mechanism in order to protect their data in within the network. This could be achieved by implementing good design of the mechanism (technical). However, it could be achieved by possessed good understanding towards the concept as well as the experience [11]. Therefore, more research which related to security (technical computation) such as data manipulation in within variety of programming and scripting languages were suggested to be conducted at large to produce more findings which then could lead to more potential solutions. Regardless of their efficiencies and significance impacts of the implementation, the research or experiment findings would at least reduce the little gap of the unknown theories in a context of computer security, and this could be done consistently and regularly.

In general, it is expected that with the good combinations of technical and available policies in relation to data protection laws (legal context) in cyber security, it would strengthen the safety of the data and would deliver “peace of mind” among the users.

ACKNOWLEDGEMENT

The main author would like to express his gratitude to the Office of Research Development & Consultancy of INTI International University (IU) and Faculty of Information Technology INTI IU for the supports in terms of resources allocation for the research as well as to Assoc. Prof. Dr. Norshima Zainal Shah and Dr. Hoo Yann Seong of National Defence University of Malaysia for the priceless supports, research advice and motivation in doing research. The authors also would like to express their gratitude to Dr. A. Selamat, former associate researcher at the Institute for Mathematical Research (UPM) for important advice on general computational logic.

REFERENCES

[1] Son, S. H. and Kang, K. D., 2002, June. QoS management in web-based real-time data services. In *Proceedings Fourth IEEE International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems (WECWIS 2002)* (pp. 129-136). IEEE.

[2] Baker, W.M., 2006. What's Your Main Technology Concern?. *Strategic Finance*, *88*(6), p.48.

[3] Biswa, K., Jamatia, B., Choudhury, D. and Borah, P., 2016. Comparative analysis of C, FORTRAN, C# and Java programming languages. *Int J Comput Sci Inf Technol*, *7*(2), pp.1004-7.

[4] Mirakhorli, M., Galster, M. and Williams, L., 2020. Understanding Software Security from Design to Deployment. *ACM SIGSOFT Software Engineering Notes*, *45*(2), pp.25-26.

[5] Jeffries, R., Turner, A.A., Polson, P.G. and Atwood, M.E., 1981. The processes involved in designing software. *Cognitive skills and their acquisition*, *255*, p.283.

[6] Santos, J.C., Peruma, A., Mirakhorli, M., Galster, M., Vidal, J.V. and Sejfia, A., 2019. Understanding Software Vulnerabilities Related to Architectural Security Tactics.

[7] Sommerville, I., 2011. Software engineering 9th Edition. ISBN-10, 137035152.

[8] Burk, P. L., 2000. Jammin'on the Web-a new Client/Server Architecture for Multi-User Musical Performance. In *ICMC*.

[9] Spillner, A., Linz, T. and Schaefer, H., 2014. *Software testing foundations: a study guide for the certified tester exam*. Rocky Nook, Inc. pp.9- 41

[10] Graham, D., Van Veenendaal, E. and Evans, I., 2008. *Foundations of software testing: ISTQB certification*. Cengage Learning EMEA.

[11] Adelson, B. and Soloway, E., 1985. The role of domain expenence in software design. *IEEE Transactions on software engineering*, (11), pp.1351-1360.