A NEW APPROACH "RING" FOR RESTRICTING WEB PAGES FROM SCRIPT ACCESS

Binayak Panda[1], K Murali Gopal[2], Pradeepta Kumar Panigrahi[3]

[1]Asst. Prof., Dept. Of IT, GIET, GUNUPUR, Odisha, India, ***binayak.panda@gmail.com***
[2]Assoc. Prof., Dept. Of CSE, GIET, GUNUPUR, Odisha, India, ***kmgopal@giet.edu***
[3]Asst. Prof., Dept. Of IT, GIET, GUNUPUR, Odisha, India, ***pradeeptapanigrahi@gmail.com***

Abstract

Web Pages which are meant for user registration or Comment in Blogs or Online Polls or logging in to a web portal or getting indexed in search engines are open to be accessed by scripts. It means it cannot be known whether the page being requested from a Human user or from a machine. There are different Websites who provides specific API to be used by scripts, for example: API for sending SMS through script. But there may be situations where a web page may need to be restricted from script access i.e. user must be a Human Being. Few popular techniques for restricting the web page from script access are "use of CAPTCHA" or "use of OTP". In this paper a different approach named as "RinG: Random id name Generator" is proposed and then the popularly used approaches have been compared with different parameters.

Keywords: *Captcha, OTP (One time Password) , RinG , API.*

----------***----------

1. INTRODUCTION

Web pages are HTML files processed by a web browser generated by a web server. An interactive web page will be consisting of a form, where the form will have different elements with specific id. While submitting the form to the server as a submission action all the ids and their respective values are being supplied either using GET or POST method. As all those ids are fixed they can be collected and the respective URL can be formed and submitted to the web server through a script.

Example:--
The home page let us say "home.php"

```
<html>
<head>
<title>RESTRICTINGSCRIPT
ACCESS</title>
</head>
<body>
<form name="sample"
id="sample" method="GET"
action="action.php">
<input type="text"
name="NUM1" id="NUM1"
value=""/>
<input type="text"
name="NUM2" id="NUM2"
value=""/>
<input type="submit"
value="ADD" />
</form>
</body>
</html>
```

The action page let us say "action.php"

```
<?php
$x=$_GET['NUM1'];
$y=$_GET['NUM2'];
echo $x+$y;
?>
```

For this example within a script the URL can be created and used as bellow:

http://localhost/paper/action.php?NUM1=12&NUM2=13
Where it cannot be predicted whether a human user is requesting for the URL or a script is requesting.

Using the above stated approach a script can do the user registration, Commenting in Blogs, Vote in Online Polls, Logging in to a web portal and also get indexed in search engines.

But many a time the web pages are meant to be accessed by human user only. To achieve the same the existing and popular approaches are "use of Captcha" or "use of OTP".

2. THE EXISTING APPROACH: CAPTCHA

A CAPTCHA (an acronym for "Completely Automated Public Turing test to tell Computers and Humans Apart") is a type of

challenge-response test used in computing to determine whether or not the user is human. The term was coined in 2003[1] by Luis von Ahn, Manuel Blum, Nicholas J. Hopper. CAPTCHAs are used to prevent bots from using various types of computing services or collecting certain types of sensitive information. Applications include preventing bots from taking part in online polls, registering for free email accounts (which may then be used to send spam) and collecting email addresses. CAPTCHAs can prevent bot-generated spam by requiring that the (unrecognized) sender pass a CAPTCHA test before the email message is delivered, but the technology can also be exploited by spammers by impeding OCR detection of spam in images attached to email messages. CAPTCHAs have also been used to prevent people from using bots to assist with massive downloading of content from multimedia websites. They are used in online message boards and blog comments to prevent bots from posting spam links as a comment or message. The different types of CAPTCAHs being used are Image based Text CAPTCHA, Audio CAPTCHA and Arithmetic CAPTCHA.[2]

Modern Text based CAPTCHAS are designed in such a way that they require the simultaneous use of three separate abilities—invariant recognition, segmentation, and parsing—to correctly complete the task with any consistency.[3]

Invariant recognition refers to the ability to recognize the large amount of variation in the shapes of letters. There are nearly an infinite number of versions for each character that a human brain can successfully identify. The same is not true for a computer and teaching it to recognize all those differing formations is an extremely challenging task. Segmentation or the ability to separate one letter from another is also made difficult in CAPTCHAs, as characters are crowded together with no white space in between. Lastly, context is also critical. A complete holistic view of the CAPTCHA must be taken to correctly identify each character. For example, in one segment of a CAPTCHA, a letter might look like a “m.” It is only when the whole word is taken into consideration that it becomes clear that it is actually a “u” and an “n.” Each of these problems poses a significant challenge for a computer even in isolation. The presence of all three at the same time is what makes CAPTCHAS so difficult to solve.[4]

3. THE EXISTING APPROACH: OTP

A one-time password (OTP) is a password that is valid for only one login session or transaction. OTPs avoid a number of shortcomings that are associated with traditional (static) passwords. The most important shortcoming that is addressed by OTPs is that, in contrast to static passwords, they are not vulnerable to replay attacks. This means that a potential intruder who manages to record an OTP that was already used to log into a service or to conduct a transaction will not be able to abuse it, since it will be no longer valid. On the downside, OTPs are difficult for human beings to memorize. Therefore they require additional technology to work.[5]

Various methods for generating OTP are used. Few are summarized below.

- A time-synchronized OTP is usually related to a piece of hardware called a security token (e.g., each user is given a personal token that generates a one-time password). Inside the token is an accurate clock that has been synchronized with the clock on the proprietary authentication server. On these OTP systems, time is an important part of the password algorithm, since the generation of new passwords is based on the current time rather than, or in addition to, the previous password or a secret key. This token may be a proprietary device, or a mobile phone or similar mobile device which runs software that is proprietary, freeware, or open-source. An example of time-synchronized OTP standard is TOTP.
- Each new OTP may be created from the past OTPs used. An example of this type of algorithm, credited to Leslie Lamport, uses a one-way function (call it f). The one-time password system works by starting with an initial seed s, then generating passwords f(s), f(f(s)), f(f(f(s))), ...as many times as necessary. Each password is then dispensed in reverse, with f(f(...f(s))...) first, to f(s). If an indefinite series of passwords is wanted, a new seed value can be chosen after the set for s is exhausted. The S/KEY one-time password system and its derivative OTP are based on Lamport's scheme.

Methods of delivering the OTP are Text messaging, proprietary tokens, Web-based methods and Paper etc.

4. PROPOSED APPROACH: RING (RANDOM ID NAME GENERATOR)

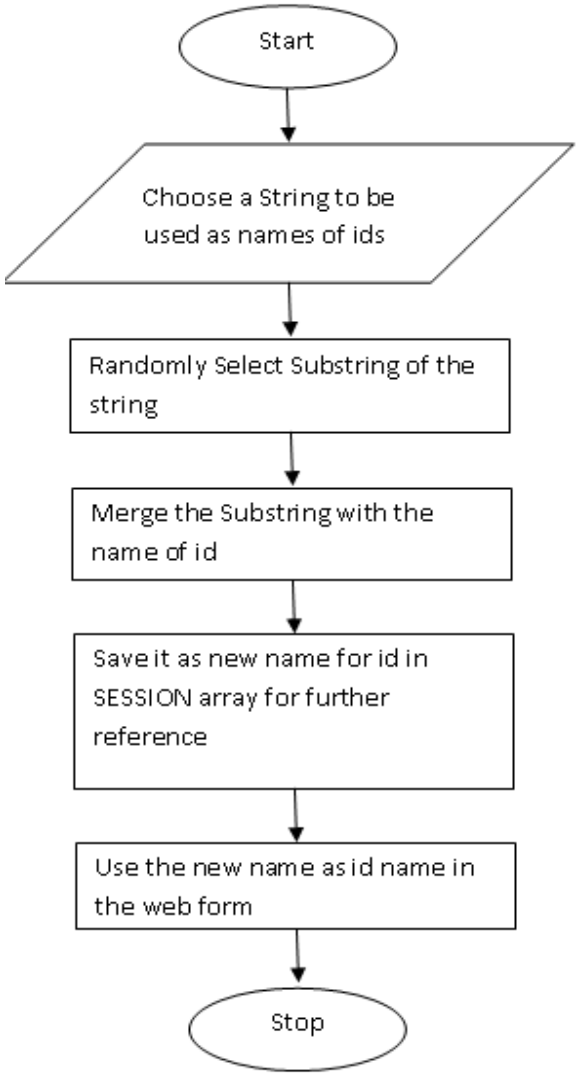
When we want to access a web page normally a request is being sent to the web server and the web server replies back with the HTML content of the desired page.

So each time the web page is requested the HTML file can be formed where the id of different element of the form can be different i.e. session specific.[6][7] As a result the URL cannot be formed by collecting the ids from the form for various elements. Even if a script collects the ids and forms the URL it will not work because of the difference in the session. Refer Flow Chart-1.

The following steps can be followed

- Let the server environment decide the name to be used as ids in the form of the requested webpage.
- Any kind of randomized concept can be used for the generation of names to be used as ids.

- Using SESSION array of the server environment let the strings meant for ids be saved on the server itself.
- When the requested web page is filled with the input values and submitted to server for some action to be carried out, let the server find the ids from session array and using them collect the submitted values from the GET/POST array and reply back with the response.



Flow chart 1

Example: --
The home page let us say “home.php”

```
<?php
session_start();

//START OF “GENERATING ID NAMES”
$X="The Brown Fox Jumping Over The Dog";
$Y=wordwrap($X,2," ",true);
$VAR=explode(" ",$Y);
$i=rand(0,16);
$NUM1="NUM1".$VAR[$i];
$_SESSION["id2"]=$NUM1;
$i=rand(0,16);
$NUM2="NUM2".$VAR[$i];
$_SESSION["id3"]=$NUM2;
//END OF “GENERATING ID NAMES”

echo "<html>";
echo "<head>";
echo "<title>RESTRICTING SCRIPT ACCESS
        </title>";
echo "</head>";
echo "<body>";
echo "<form name=\"addition\" id=\"addition\"
        method=\"GET\" action=\"action1.php\">";
echo "<input type=\"text\" name=\"$NUM1\"
        id=\"$NUM1\" value=\"\"/>";
echo "<input type=\"text\" name=\"$NUM2\"
        id=\"$NUM2\" value=\"\"/>";
echo "<input type=\"submit\" value=\"SUM\" />";
echo "</form>";
echo "</body>";
echo "</html>";
?>
```

The action page let us say “action1.php”

```
<?php
session_start();

//START OF “EXTRACTING ID NAMES”
$RECEIVED=array();
$RECEIVED[]=$_SESSION["id2"];
$RECEIVED[]=$_SESSION["id3"];
//END OF “EXTRACTING ID NAMES”

$i=0;$sum=0;
while(true)
{
        if(array_key_exists($i,$RECEIVED))
        {
                $k=$RECEIVED[$i];
                $sum+=$_GET[$k];
                $i++;
        }
        else
        {
        break;
        }
}
echo $sum."<br><br>";
session_destroy();
?>
```

In the above said approach each time the home page gets submitted the URL gets changed.

Test Case1:
http://localhost/paper/action1.php?NUM1Ov=12&NUM2ow=13

Test Case2:
http://localhost/paper/action1.php?NUM1Br=12&NUM2ow=13

5. A COMPARISON OF THE PROPOSED APPROACH WITH THE EXISTING APPROACHES

Here the proposed approach is compared with the existing approaches. In the following table the comparison has been carried out considering 8 numbers of parameters.

Table -1: Comparison of All Approaches

Approach /Parameters	Approach1 (Captcha)	Approach2 (OTP)	Approach3 (RinG)
Involvement of cost (Vendor)	YES	YES	NO
Input Validation	YES	YES	YES
Browser Dependency	YES	NO	NO
Secured	YES	YES	YES
Dependence on Internet SPEED	YES	YES	YES
Dependence on Server Configuration (Speed & Memory)	YES	YES	YES
Dependence on Client Configuration (Memory)	YES	YES	NO
Web page access time (X time unit)	1.5 X	1.5 X	X

CONCLUSIONS

In this paper the proposed approach has been proven to be an efficient, easy and effective approach for restricting the web pages from script access. It is cost effective also in terms of money, Processing Speed, Security as there is no involvement of Vendor cost as like other two approaches. In this paper a static string has been chosen for generation of id names. It can be extended where by using a randomized algorithm for generation of the string to be used for id name generation which will minimize the id name predication hence by improving security.

REFERENCES

[1]. Ahn, Luis von; Blum, Manuel; Hopper, Nicholas J.; Langford, John (2003). "CAPTCHA: Using Hard AI Problems for Security". Advances in Cryptology — EUROCRYPT 2003. Lecture Notes in Computer Science 2656. pp. 294–311. doi:10.1007/3-540-39200-9_18. ISBN 978-3-540-14039-9

[2]. http://en.wikipedia.org/wiki/CAPTCHA

[3]. Chellapilla, Kumar; Kevin Larson, Patrice Simard and Mary Czerwinski. "Designing Human Friendly Human Interaction Proofs (HIPs)". Microsoft Research.

[4]. Bursztein, Elie; Matthieu Martin, and John C. Mitchell (2011). "Text-based CAPTCHA Strengths and Weaknesses". ACM Computer and Communication security 2011 (ACM Conference CSS'2011). Stanford University.

[5]. http://en.wikipedia.org/wiki/One-time_password

[6]. PHP : The Complete Reference 1st Edition, By Steven Holzner, Publisher: Tata McGraw - Hill Education ISBN-9780070223622

[7]. Beginning PHP6, Apache, MySQL Web Development by Timothy Boronczyk Elizabeth Naramore Jason Gerner Yann Le Scouarnec Jeremy Stolz Michael K. Glass, Publisher:Wiley India Pvt Ltd, ISBN 9788126521227

BIOGRAPHIES

Prof. Binayak Panda has received a bachelor's degree in Computer Science and Engineering from BPUT Odisha in the year 2005. In the year 2010 he has received a master of technology degree in Computer Science and Engineering from BPUT Odisha. Currently he is working as an Asst. Prof. in the Dept. of IT at GIET Gunupur. He has 3 years of industry experiences in the field of Software testing and maintenance. His interested areas of research are Software Engineering and Real Time System. He is a life time member of ISTE.

Prof. K Murali Gopal has received a bachelor's degree in Computer Science and Engineering from BPUT Odisha in the year 2003. In the year 2007 he has received a master of technology degree in Computer Science and Engineering from BPUT Odisha. Currently he is working as an Assoc. Prof. In the Dept. of CSE at GIET Gunupur. He has 2 years of industry experiences in the field of Software Analysis. His interested areas of research are Software Engineering and Computer Vision. He is a life time member of ISTE.

Prof. Pradeepta Kumar Panigrahi received double master degree in MCA and M.Tech from BPUT Odisha. Currently he is working as an Asst. Prof. in the Dept. of IT at GIET Gunupur. He has 3 years of industry experiences in IBM India and TCS in the field of development, Software testing and maintenance. His interested areas of research are Software Engineering and Real Time System. He is a life time member of ISTE.