Journal of Computers Vol. 28, No. 5, 2017, pp. 62-68
doi:10.3966/199115992017102805006

The Research and Improvement in the Detection of PHP Variable WebShell based on Information Entropy

Chundong Wang[1], Hong Yang[1*], Zhentang Zhao[1],
Liangyi Gong[1] and Zhiyuan Li[1]

[1] School of Computer and Communication Engineering, Tianjin University of Technology Tianjin, China 300384
{michael3769, youngTJUT}@163.com

Received 20 June 2016; Revised 7 February 2017; Accepted 8 February 2017

Abstract. The In recent years, a trend to implant into the back door for website attack has been increasing, using back door to tamper the application system, stealing the sensitive information in database and cause great threat. The existing technology to Webshell backdoor detection method is generally static attributes, can search and kill common backdoor attack, but because of the variable WebShell often disguised as a normal WEB script file, this kind of dynamic behavior detection technology often difficult to handle, can not effectively detect variable WebShell. In order to detect variable WebShell, we propose an information entropy detection algorithm based on PHP special strings, use a normal file information entropy for threshold, detect whether the PHP file contains Webshell or not. On this basis, in order to slove difficulties with non-ASCII code and digital variable WebShell as well as the flexibility of the PHP language dynamic function, we propose detection algorithm based on quotation information entropy. The experimental results shows that special string information entropy detection algorithm based on PHP and detection algorithm based on quotes entropy can better detect variable Webshell with high accuracy and low false alarm rate. The PHP special string detection algorithm based on information entropy with detection algorithm based on quotation information entropy show better detection performance in handle with difficulties to detect ASCII and digital variable Webshell.

Keywords: characteristic value detection, information entropy, PHP, variable Webshell

1 Introduction

With the continuous development of information technology, information system coveys not only traditional functions like news release, content exhibition, but also undertake Information query, order processing, transaction management, etc. which often involves some information need to be kept in secret, Web application script back door arises at the historic moment. After the success of the hackers in the use of WEB application vulnerabilities, they usually make use of implant WellShell backdoor to tamper the information system [8], control the operating system and steal the sensitive data from database. The attacker use browser and control site to exchange data with be controlled site through legal ports, which stays in high concealment so that traditional firewall hardly to intercept and no operating record in the system log.

Some special functions is inevitable because of the need to complete the use of some special functions. We use these characteristics to locate WebShell [9]. The existing technology for Webshell backdoor detection method is generally static attributes, it can search and kill common backdoor attack. But the variable WebShell backdoor tend to simulate normal operation for database, they do not have an obvious static special properities, and can not form relatively obvious access features because of its fewer accessed time, so the values of the general detection cannot be effective for variable WebShell [11].

[*] Corresponding Author

Scholars have done a lot of research on webshell detection. Du puts forward a webshell detection method based on PHP extension, this method can real-time detect the webshell's running [1]. Ye proposes a black box detecting method based on support vector machine classification algorithm by analyzing the HTML feature of Webshell pages [2]. Kang puts forward a method from the php kernel based on the use of hook technology to achieve a php-based expansion of the webshell defense mechanism [3]. Hu proposes a kind of detection model based on naive Bayesian classification theory, this model can effectively detect the confused Webshell [4]. However, due to the flexibility of the PHP language dynamic function, the above research methods are not effective

Based on the flexibility of the PHP language dynamic function, we propose an information entropy detection algorithm based on PHP special strings. This algorithm select special characters in files as test objects, take normal file information entropy as the standard threshold then detect information entropy in the file and compare to threshold to determine whether a file contains variable WebShell script. As for non-ASCII code and digital variable Webshell [13], we propose quotes entropy detection algorithm use its characteristics of quotes, in order to decide whether the file contains no ASCII and digital variable Webshell or not, we detect quotes entropy information in the file to compare with the normal ones.

2 Webshell Detection

The meaning of "Web" is the need to open the Web services server, the meaning of "shell" is to obtain the permissions for the operation of the server in a certain level, it is often referred to as intruders degree of operation permissions in website servers through website ports [6].

Take the executed command for example, Webshell can invoke the internal function of the script file to perform the operating system shell commands as long as enough permissions. For example, the following statement:

```
<?php echo system($_GET['cmd'];?>
```

There are often seen a word WebShell, which usually used to execute script code passed by clients. For example, the following statement:

```
<?php eval($_POST['code'];?>
```

If this common Webshell want to perform a specific function, special functions and variables must be needed, those special functions are distinguished features for detection. Such as system, eval, GET, POST .ect [12]. Webshell use methods like searching and killing, encrypted, transformation, call-back, hide the keywords to hide its characteristics, then turn into data section ($_GET) and transfer data ($_POST), execute php fundamental function like exec(), eval(), so that we can not judge whether this script is illegal or not. For example, the script below is a none ASCII code and digital variable Webshell.

```
<?php
$_="                                                                  ";
$_[+"                              "]='                               ';
$_="$_"."                                                             ";
$_=($_[+"        "]|"0x06").($_[+"        "]|"0x05").($_[+"      "]    ^    "0x15");
?>
```

Where, $_=($_[+" "]|"0x06").($_[+" "]|"0x05").($_[+" "] ^ "0x15"); can get the result of GET.

2.1 The Common Method for Webshell Detection

Match with characteristic values. Match with characteristic values is the most commonly method in detection. We usually make a collection for Webshell characteristics fingerprint and consider it as a database to be matched with. If the feature are match, then the file is Webshell, but variable Webshell can evade it [10].

Overlap index detection. Software development use a unification of function and variable names write regularly to compile. But Webshell is different from Web, Webshell has a low overlap index of WEB application code, so if the overlap index is low, than more likely to be encrypted and more likely to be Webshell [5].

The longest word in the file. Ask how long is the longest word in a file, more unusually long words is very suspicious, PHP usually use base64 for encoding, then call the function to decode, in order to avoid

WAF and other tools detection, so if the length of the file exceed a certain limit, than the file is more likely a Webshell [14].

The existing detection technology of Webshell usually contains more obvious static characteristics or behavior patterns, can not make effective detection for variable Webshell, Due to the flexibility of the PHP language itself, the behaviors such as support function dynamic execution or function name can be dynamically generated and run in the program execution environment can cause some string functions dynamically generated special executive function name, thus can dynamically generate functions and implement dangerous code. Methods based on dynamic behavior detection technology often difficult to realize and influence the performance of the system, may even affect system stability [7].

3 The Research and Implementation of PHP Variable WebShell Detection Algorithm based on Information Entropy

The information entropy is a rather abstract concepts in mathematics, here take the information entropy as a probability for the appearance of particular information (The emergence of discrete random event probability). If the system is in order than the information entropy is low, on the other hand, if the system is disorder, than the information entropy is high, information entropy can defined as a metric of ordering in a system [8, 15].

Entropy is considered from the statistical features of the whole set, from an average of meaning it represent the general characteristics of information source, the formulas are as follows:

$$H(x)=E\left[\log\frac{1}{p(a_i)}\right]. \quad \textbf{(1)}$$

If system contains several events $S=E_1,\ldots,E_n$, each event probability distribution $p=p_1,\ldots,p_n$, Information of each event itself $I_e=-\ln p_i$, Then plug in the above formula, get the information entropy value of the random file as threshold, get the information entropy of each file, the higher the value, the greater the possibility for WebShell.

It will bring a lot of unreliability and uncertainty because of the information entropy is based on the random events, In this paper, we use normal file instead of random events and optimize document processing, then ask the information entropy value and as a threshold comparing with general file information entropy, if it is bigger than the threshold, the possibility of Webshell is higher.

Get the information entropy value of the normal files, obtain a non-character string length a, non-Chinese character string length b, non-blank string length k and quotation number m. Get m as the denominator, $p(a)=a/k$, $p(b)=b/k$, $p(m)=m/k$.

The expectation for special characters in normal file is $E(x_1)$, the expectation for Normal file quotes is $E(x_2)$.

$$E(x_1)=i\cdot p(i)+j\cdot p(j). \quad \textbf{(2)}$$

$$E(x_2)=m\cdot p(m). \quad \textbf{(3)}$$

At the same time record the location of the file, Then do accumulation of entropy $H(x_1)$ and $H(x_2)$ in the whole incident, the entropy as the threshold to check other files.

$$E(x_1)=-\big(p(a)\log p(a)+p(b)\log p(b)\big). \quad \textbf{(4)}$$

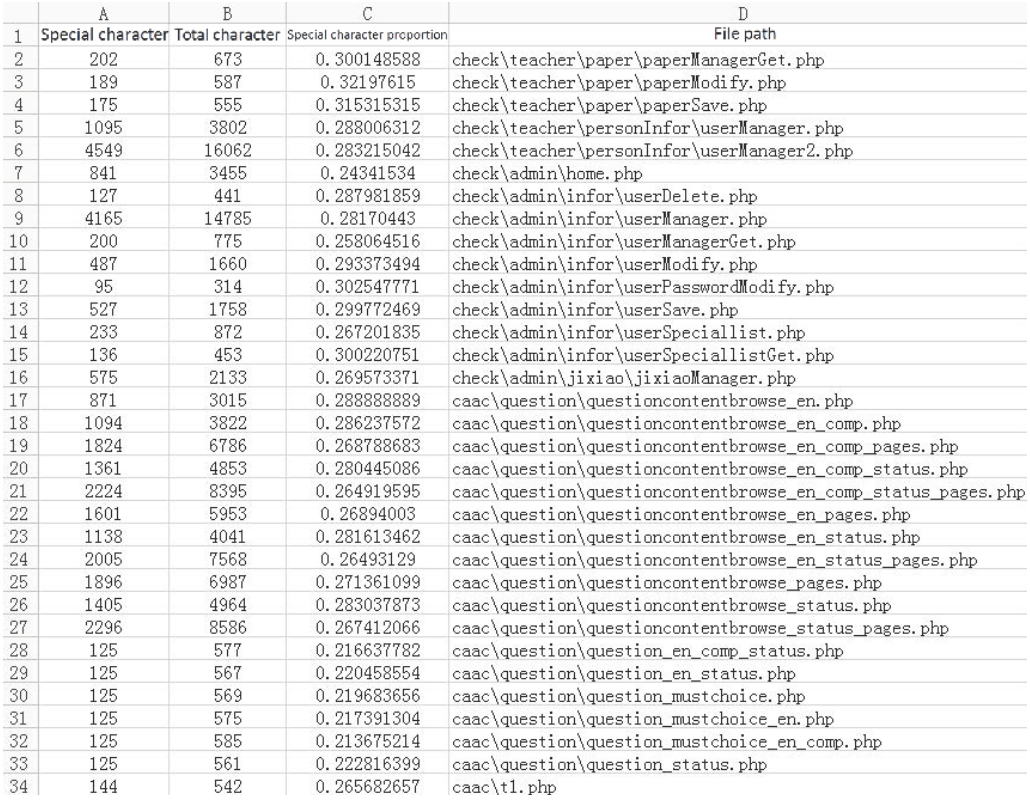
$$E(x_2)=-p(m)\log p(m) \quad \textbf{(5)}$$

Using the same method to check the information entropy value of the file H(x), compare it with $H(x_1)$ and $H(x_2)$, if $H(x)>H(x_1)$, $H(x)>H(x_2)$, then the influenza virus file may be variant WebShell. Webshell detection code is as follows:

```
 program Information_entropy_detection (file_right, file_detected,
Output)
  var a, b, k, m;
  {a=Non character string length;
   b=non Chinese character string length;
   k=non space string length;
   m=number of quotation marks};
  begin
    Pa=a/k;
    Pb=b/k;
    Pm=m/k;
    Ex1=-(Pa*log(Pa)+Pb*log(Pb));
    Ex2=-Pm*log(Pm);
    {Use the same method to get the information entropy of the
  inspection file (Ex)};
    if(Ex>Ex1 & Ex>Ex2)
       Output=variant_WebShell;
  end.
```

4 Experiment

In order to judge whether the file is normal or not, the experiment compare two algorithms at the same time. The random samples in the experiments are selected for normal files, determine the information entropy value of the normal files, then figure out the information entropy of files which to be checked. Take the information entropy in normal files as a threshold, then detect variable Webshell. A part of normal development of files are selected in Fig. 1.

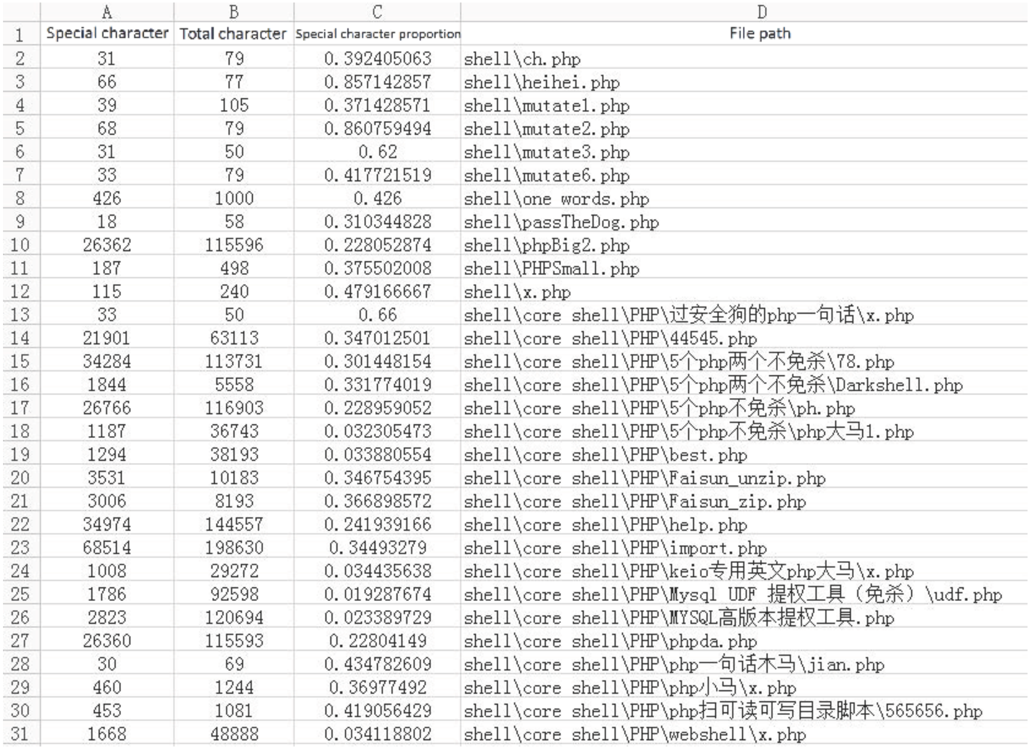


	A	B	C	D
1	Special character	Total character	Special character proportion	File path
2	202	673	0.300148588	check\teacher\paper\paperManagerGet.php
3	189	587	0.32197615	check\teacher\paper\paperModify.php
4	175	555	0.315315315	check\teacher\paper\paperSave.php
5	1095	3802	0.288006312	check\teacher\personInfor\userManager.php
6	4549	16062	0.283215042	check\teacher\personInfor\userManager2.php
7	841	3455	0.24341534	check\admin\home.php
8	127	441	0.287981859	check\admin\infor\userDelete.php
9	4165	14785	0.28170443	check\admin\infor\userManager.php
10	200	775	0.258064516	check\admin\infor\userManagerGet.php
11	487	1660	0.293373494	check\admin\infor\userModify.php
12	95	314	0.302547771	check\admin\infor\userPasswordModify.php
13	527	1758	0.299772469	check\admin\infor\userSave.php
14	233	872	0.267201835	check\admin\infor\userSpeciallist.php
15	136	453	0.300220751	check\admin\infor\userSpeciallistGet.php
16	575	2133	0.269573371	check\admin\jixiao\jixiaoManager.php
17	871	3015	0.288888889	caac\question\questioncontentbrowse_en.php
18	1094	3822	0.286237572	caac\question\questioncontentbrowse_en_comp.php
19	1824	6786	0.268788683	caac\question\questioncontentbrowse_en_comp_pages.php
20	1361	4853	0.280445086	caac\question\questioncontentbrowse_en_comp_status.php
21	2224	8395	0.264919595	caac\question\questioncontentbrowse_en_comp_status_pages.php
22	1601	5953	0.26894003	caac\question\questioncontentbrowse_en_pages.php
23	1138	4041	0.281613462	caac\question\questioncontentbrowse_en_status.php
24	2005	7568	0.26493129	caac\question\questioncontentbrowse_en_status_pages.php
25	1896	6987	0.271361099	caac\question\questioncontentbrowse_pages.php
26	1405	4964	0.283037873	caac\question\questioncontentbrowse_status.php
27	2296	8586	0.267412066	caac\question\questioncontentbrowse_status_pages.php
28	125	577	0.216637782	caac\question\question_en_comp_status.php
29	125	567	0.220458554	caac\question\question_en_status.php
30	125	569	0.219683656	caac\question\question_mustchoice.php
31	125	575	0.217391304	caac\question\question_mustchoice_en.php
32	125	585	0.213675214	caac\question\question_mustchoice_en_comp.php
33	125	561	0.222816399	caac\question\question_status.php
34	144	542	0.265682657	caac\t1.php

Fig. 1. Part of the normal file data

We can figure out the value of normal file special characters information entropy is about 0.32, normal file quotation information entropy is about 0.085.

Due to the minority in variation of Webshell, statistics the variation of Webshell samples and the normal ones when doing the experiments, using the above algorithm figure out the information entropy value of the examined files $H(x)$ and compare it with the threshold, if the value is higher than the threshold, then sound a warning, if the value is less than the threshold, we ignore it. A part of files to be examined are selected as Fig. 2.



	A	B	C	D
1	Special character	Total character	Special character proportion	File path
2	31	79	0.392405063	shell\ch.php
3	66	77	0.857142857	shell\heihei.php
4	39	105	0.371428571	shell\mutate1.php
5	68	79	0.860759494	shell\mutate2.php
6	31	50	0.62	shell\mutate3.php
7	33	79	0.417721519	shell\mutate6.php
8	426	1000	0.426	shell\one words.php
9	18	58	0.310344828	shell\passTheDog.php
10	26362	115596	0.228052874	shell\phpBig2.php
11	187	498	0.375502008	shell\PHPSmall.php
12	115	240	0.479166667	shell\x.php
13	33	50	0.66	shell\core shell\PHP\过安全狗的php一句话\x.php
14	21901	63113	0.347012501	shell\core shell\PHP\44545.php
15	34284	113731	0.301448154	shell\core shell\PHP\5个php两个不免杀\78.php
16	1844	5558	0.331774019	shell\core shell\PHP\5个php两个不免杀\Darkshell.php
17	26766	116903	0.228959052	shell\core shell\PHP\5个php不免杀\ph.php
18	1187	36743	0.032305473	shell\core shell\PHP\5个php不免杀\php大马1.php
19	1294	38193	0.033880554	shell\core shell\PHP\best.php
20	3531	10183	0.346754395	shell\core shell\PHP\Faisun_unzip.php
21	3006	8193	0.366898572	shell\core shell\PHP\Faisun_zip.php
22	34974	144557	0.241939166	shell\core shell\PHP\help.php
23	68514	198630	0.34493279	shell\core shell\PHP\import.php
24	1008	29272	0.034435638	shell\core shell\PHP\keio专用英文php大马\x.php
25	1786	92598	0.019287674	shell\core shell\PHP\Mysql UDF 提权工具（免杀）\udf.php
26	2823	120694	0.023389729	shell\core shell\PHP\MYSQL高版本提权工具.php
27	26360	115593	0.22804149	shell\core shell\PHP\phpda.php
28	30	69	0.434782609	shell\core shell\PHP\php一句话木马\jian.php
29	460	1244	0.36977492	shell\core shell\PHP\php小马\x.php
30	453	1081	0.419056429	shell\core shell\PHP\php扫可读可写目录脚本\565656.php
31	1668	48888	0.034118802	shell\core shell\PHP\webshell\x.php

Fig. 2. Part of WebShell data

Export experimental results above, draw out special characters information entropy diagram and the quote information entropy diagram using MATLAB, use file identification as abscissa, information entropy as ordinate.

It can be seen from the above experiment results, the special characters information entropy diagram and quotes entropy diagram get the highest fluctuate between 350, this is because the files between 300 and 410 are Webshell files and there are non-ASCII code and digital variable WebShell files at 340, the rest of the files is a common project, as shown in Fig. 3 based on the special characters information entropy diagram could not detect the variable Webshell in an obvious way. But accurately detect variable WebShell files shown in Fig. 4 quotes entropy figure.

Can be seen from the diagram, in the normal file range, some information entropy exceeded the corresponding threshold and lead to false positives. And then by looking at the relative file we found that this file is the configuration file which contains several cities. Within the range of ordinary Webshell, file entropy value is too large because of the algorithm in the process of calculation rule out Chinese characters, so that Two kinds of information entropy and almost normal information entropy.

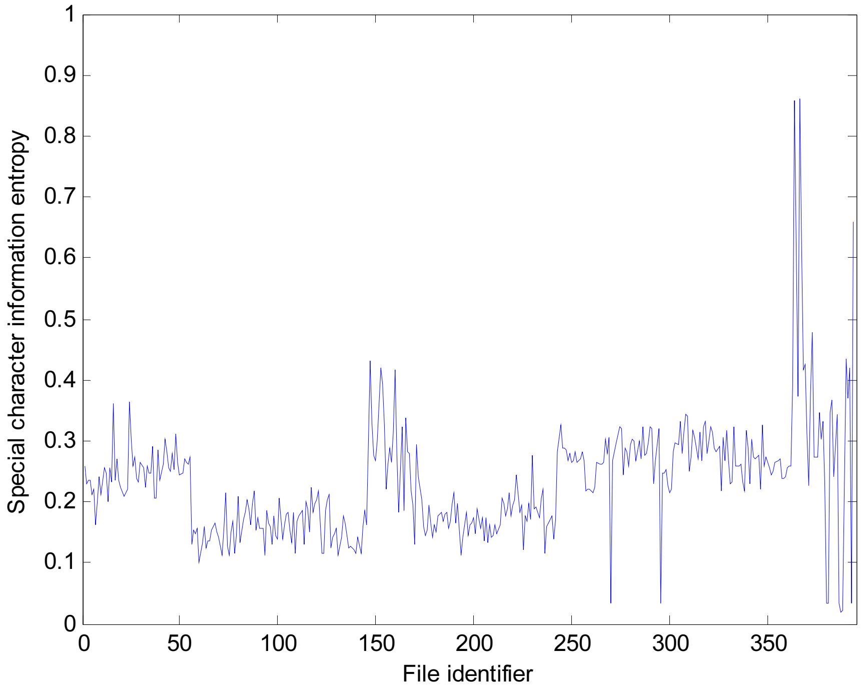


Fig. 3. Special characters information entropy diagram

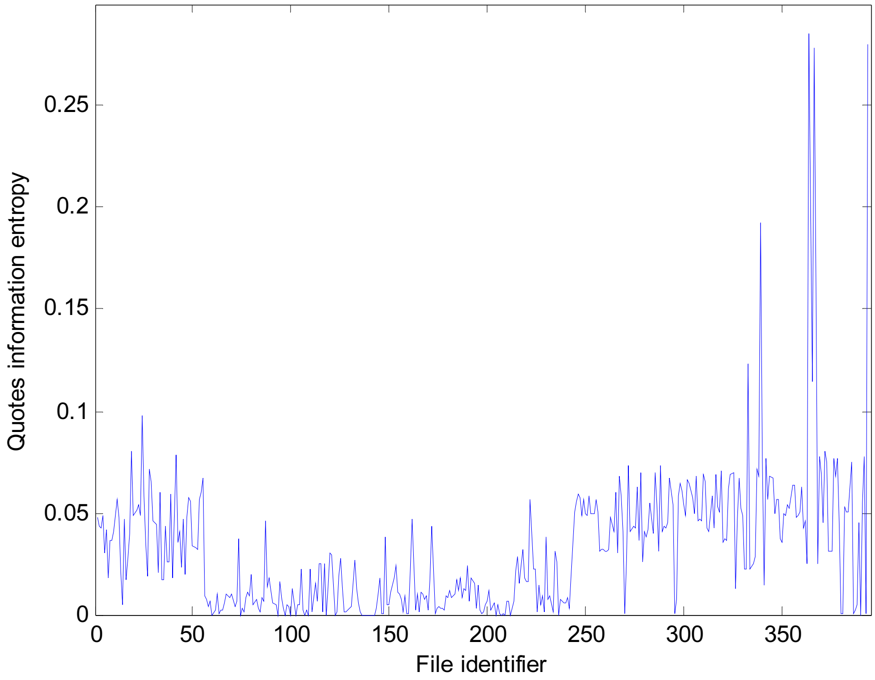


Fig. 4. Quotes entropy diagram

For Webshell detection, we usually use detection technology based on characteristic value, but it is useless for variable Webshell. In this paper, we propose the special character information entropy for PHP and quotes entropy WebShell detection algorithm, it improves the ability of detecting WebShell in a great extent. We compared this detection algorithm with traditional methods in Table 1.

Table 1. Detection algorithm compared with the traditional methods

	Based on characteristic value matching	Font size and style	Quotes entropy
Detection object	Text string	Text string	Text string
Common WebShell	Accuracy 99%, Lower error rate	medium	Low accuracy
Variable WebShell	Hardly	High accuracy	High accuracy
Non-ASCII code and	Hardly	Low accuracy	High accuracy

5 Conclusion

We have presented detection algorithms as for WebShell in PHP special characters of the information entropy and quotes entropy through WebShell detection research based on information entropy. Based on more than 400 PHP project files and one hundred kinds of WebShell detection and data statistics, Experiments prove that detection algorithm based on the special string information entropy and information entropy based on quotes detection algorithm can effectively detect the variant WebShell in high accuracy.

This article also has limitations in the number of normal file selection, the possible situation is not considered comprehensive, expand the number of normal files so that the detection threshold is more accurate, higher detection rate.

Acknowledgements

The work is supported by the Foundation of Educational Commission of Tianjin, China (Grant No. 20130801), the General Project of Tianjin Municipal Science and Technology Commission (No.15JCYBJC15600), the Major Project of Tianjin Municipal Science and Technology Commission (No.15ZXDSGX00030), and NSFC: The United Foundation of General Technology and Fundamental Research(No.U1536122).

References

[1] H. Du, Y. Fang, PHP Webshell real time dynamic detection, Network Security Technology and Application 12(2014) 120-121.

[2] F. Ye, J. Gong, W. Yang, Webshell black box detection based on support vector machine, Journal of Nanjing University of Aeronautics & Astronautics 47(2015) 924-930.

[3] Z. Kang, Research of Webshell detection based on PHP extension, Science and Technology Communication 19(2015) 123-124.

[4] B. Hu, Research on Webshell detection method based on Bayesian theory, Science and Technology Square 6(2016) 66-70.

[5] J. Cao, B. Yu, F. Dong, X. Zhu, S. Xu, Entropy-based denial-of-service attack detection in cloud data center, Concurrency and Computation: Practice and Experience 27(18)(2015) 5623-5639.

[6] L.Y. Deng, D.L. Lee, Y.H. Chen, L.X. Yann, Lexical analysis for the webshell attacks, in: Proc. 2016 International Symposium on Computer, Consumer and Control (IS3C), 2016.

[7] Q. Jianjun, Detection method of stealth webshell, Computer and Network 13(2015) 38-39.

[8] J. Kim, D.H. Yoo, H. Jang, K. Jeong, Webshark 1.0: a benchmark collection for malicious web shell detection, Journal of Information Processing Systems 11(2)(2015) 229-238.

[9] X. Mingkun, C. Xi, H. Yan, Design of software to search asp web shell, Procedia Engineering 29(2012) 123-127.

[10] X. Mo, W.C. Chang, Web application intrusion detection model based on active entropy, Journal of Wuhan University 60(6)(2014) 543-547.

[11] T.D. Tu, C. Guang, G. Xiaojun, P. Wubin, Webshell detection techniques in web applications, in: Proc. Computing, Communication and Networking Technologies (ICCCNT), 2014.

[12] P. Wrench, B. Irwin, Detecting derivative malware samples using deobfuscation-assisted similarity analysis, SAIEE Africa Research Journal 107(2)(2016) 65-77.

[13] P.M. Wrench, B.V. Irwin, Towards a php webshell taxonomy using deobfuscation-assisted similarity analysis, in: Proc. Information Security for South Africa (ISSA), 2015.

[14] H. Wu, X. Dang, L. Wang, L. He, Information fusion-based method for distributed domain name system cache poisoning attack detection and identification, IET Information Security 10(1)(2016) 37-44.

[15] M. Xu, X. Chen, Y. Hu, Design of software to search ASP Web Shell, Procedia Engineering 29(2012) 123-127.