INFORMATION
SYSTEMS IN
MANAGEMENT

Information Systems in Management (2015) Vol. 4 (1) 14−25

VISUALIZATION AS SUPPORT FOR WEB HONEYPOT DATA ANALYSIS

KRZYSZTOF CABAJ

Institute of Computer Science, Warsaw University of Technology

The paper presents methodologies associated with visualization, which supports data analysis. Analyzed data has been gathered by HoneyPot systems deployed in the network of Institute of Computer Science. Due to the vast amounts of data, the manual analysis was almost impossible and very impractical, also considering time constraints. Introduced visualization techniques and supporting filtering features are implemented in HPMS (HoneyPot Management System). The paper describes in details two introduced methodologies which support data analysis using both charts and graphs. The first one is used for the discovery of basic activities observed by HoneyPot. The second one is used for advanced analysis of machines used during attacks concerning PhpMyAdmin software.

Keywords: HoneyPot systems, visualization, data-mining, monitoring

1. Introduction

In the „Big data" days, the need for analysis of data gathered in vast amounts, by various information systems, cannot be questioned. For this purpose, multiple methods can be used, starting from the simplest one, implemented in each modern data base system, which allows computation of average, minimal, maximal or count of given group of records, ending with the most advanced one that utilizes data mining techniques which could automatically group given data or discovered patterns appearing in the analyzed data. Although, described methods decrease the number of data presented, rapid discovery of most important information could be very difficult and time consuming. Most often the new approach is used, which utilizes the visualization of data. The good visualization, even of vast amounts of data, allows easy discovery of most important relations or rules.

In the paper results and conclusions concerning conducted experiments associated with the analysis of HoneyPot logs are presented. The HoneyPot system is a kind of a trap for attackers, and data from such a system are gathered by the author since a few years now. Nowadays, vast amounts of such data are acquired, even dozen thousands events a month. For the analysis purpose, a dedicated system called HPMS (Honey Pot Management System) [1] was developed at the Institute of Computer Science, Warsaw University of Technology. The first implemented methods utilized simple data base functions which, for example, allow discovery of most active machines. During the following research, advanced data mining techniques that discover patterns are used, namely frequent sets and jumping emerging patterns, were designed. However, even using these methods a discovery of trends and associations between events was problematic. For this purpose the HPMS system has been extended with modules which allow data visualization. Currently, modules for plotting significant parameters during time and associations between various events using graphs have been implemented. The usage of such methods allows rapid discovery of time events, when new activity emerges or connections between them are discovered. In effect, the analytic work can be efficient, and reaction to new trends can be implemented.

The paper is organized as follows. The next, second, section describes HoneyPot systems, especially those which are used for gathering data for further analysis. The third section is associated with analyses that are supported by plots generated using HPMS systems. The fourth section describes the analysis of attacks directed to the PhpMyAdmin vulnerability using graphs. The last, fifth section concludes the conducted experiments.

2. HoneyPot systems

The role of the HoneyPot can be performed by any resource which can be used for observing hostile or unexpected activity. The only common feature of this resource is that it is not used for production purposes. The HoneyPot is usually a specialized machine or software; however, this role can be taken by a fake record in the data base or the account in the important computer system. Any access to the resource, for example, an attempt to read or login, is a sign of an unexpected activity. Historically, specially configured computers were used as the HoneyPot system. The configurations enabled various monitoring mechanisms that gathered as many data as possible concerning the attacker activity during an attack. For this purpose, logs from operating systems, logs from network devices placed between HoneyPot and Internet or even traces of all traffic directed to it can be used. Such a solution was ideal for catching and tracking a human attacker, but it also had many drawbacks. The first and the most important one is associated with an additional risk. If the attacker detects and disables all the monitoring mechanism, the HoneyPot can be used for other hostile activity. Additionally, the initial deployment or cleaning the HoneyPot after a successful attack is very labor intensive. This kind of systems are called high interaction HoneyPots. In [2], the details concerning one of the first well documented development of the HoneyPot system and the description of further monitoring and tracing a real attacker can be found.

In the era of automatic threats, like worms, e-mail viruses or auto-rooters, dedicated high interaction HoneyPot systems used for gathering copies of new samples of malicious code are inefficient and very risky. After each infection the HoneyPot system must be cleaned. This process is relatively slow, even with the support of virtualization. A better solution for gathering information related to malware is the usage of low interaction HoneyPots. The low interaction HoneyPot is a dedicated software which imitates vulnerable services. Depending on the purpose, it can be very simple, for example, only listing for incoming connections and returning standard banners of a simulated service on the one hand. On the other hand, there are very complicated systems dedicated to downloading new samples of malware. This kind of low interaction HoneyPots simulates high level protocols in which vulnerabilities appear, incoming shellcode used by worm during vulnerability exploitation is emulated and the next stages of the activity of malware is downloaded. The most important low interaction HoneyPots are HoneyD [3], Nepenthes [4] and its successor Dionaea [5].

During our research on automatic threats, conducted at the Institute of Computer Science, only low interaction HoneyPots were used. Due to limitations of available systems, associated with very poor simulation of Web applications, a custom solution named WebHP was introduced. More details concerning its functionality are presented in papers [1, 6].

3. Analysis of HoneyPot data using charts

The WebHP HoneyPot system described in the previous section produces vast amounts of data that are later analyzed. For example, during the one and a half year the WebHP system receives more than 300 thousands of connections. Manual identification of new attacks or trends detection is almost impossible without dedicated software. Moreover, the analysis using only text representation of each connection is a tedious work. The most straightforward way to visualize this data is the usage of a chart. The Fig. 1 shows a sample chart from HPMS systems, which presents the activity observed by WebHP sensors during the year 2013. The X-axis represents time and the Y-axis represents number of connections observed during each day.

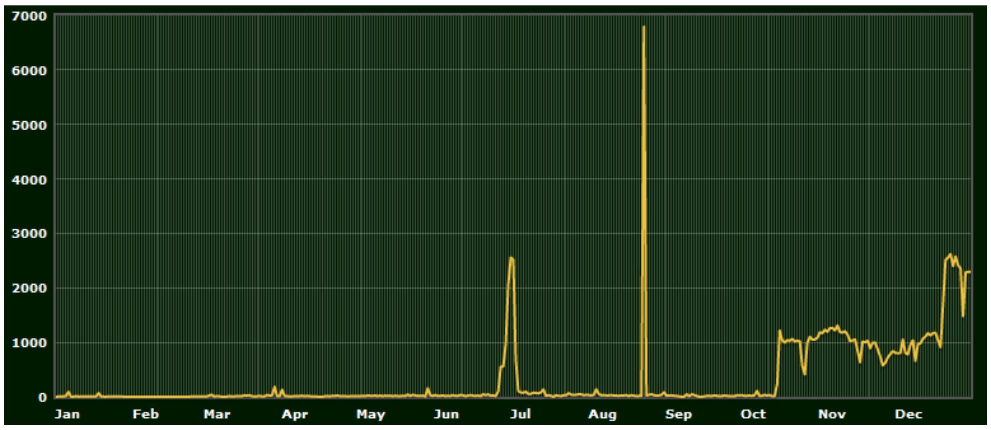


Figure 1. Activity observed by WebHP sensors during the year 2013. Chart generated by HPMS software

At the first glance, a few very interesting time instants and time ranges can be identified. This information can be beneficial for a person which performs the analysis and points out the time range which should be investigated at first. In the presented Fig. 1, the sharp spike is associated with execution of security scanner. Both time ranges with high activity, the first between the end of June and middle of July and the second from October to the end of December, are related to automatic SPAM activity which placed marketing posts at the insecure webpage of the guest book. More details concerning this attack are presented in the article [1].

For more detailed analysis, the charts can be plotted for any tag associated with the connection. Tag definition is configured by the user, and later each connection which fulfill tags' constraints is automatically marked by the given tag. Tag definition has a form of a logical expression which additionally can use any field appearing in the HTTP header. Fig. 2 shows sample logical expressions which are used for connection tagging.

```
header='[POST] configuration' AND (value ILIKE '%42.php%'
OR value ILIKE '%43.php%' OR value ILIKE '%44.php%' OR
value ILIKE '%45.php%' OR value ILIKE '%124.php%')
```

Figure 2. Sample tag definition used for automatic connection marking in the HPMS system

Using charts and tags provided by the HPMS system methodology for data analysis was introduced. At the first step the person which performs analysis for a given time period observes the initial chart. Interesting features of the chart, for example, sharp spikes or time ranges which shows an observable rise in HoneyPot activity, are detected and later investigated in details. In many cases a detailed investigation reveals knowledge associated with detected feature in the chart, for example IP address of the offending machine or a repeated pattern in a field. In most cases using this information new tag can be created. In the next step a new chart, without previously detected transactions, is generated. The work is performed until there are no transactions which can be identified and marked.

The described methodology can be applied to data presented in the Fig. 1. Identified attacks lead to defining two tags that can be used for data filtering. Fig. 3 shows the activity of the deployed HoneyPot system in 2013 without these two attacks. As it can be seen, the usage of these two tags reduced the number of transactions almost ten times. In the new chart, other spikes can be observed. As was introduced in the methodology, these spikes should be analyzed and in effect new tags can be produced and additional knowledge associated with observed attacks can be gathered. Despite the fact that activity level in Fig. 3 compared to the Fig. 1 is one or two order of magnitude lover, methodology of analysis is almost the same.

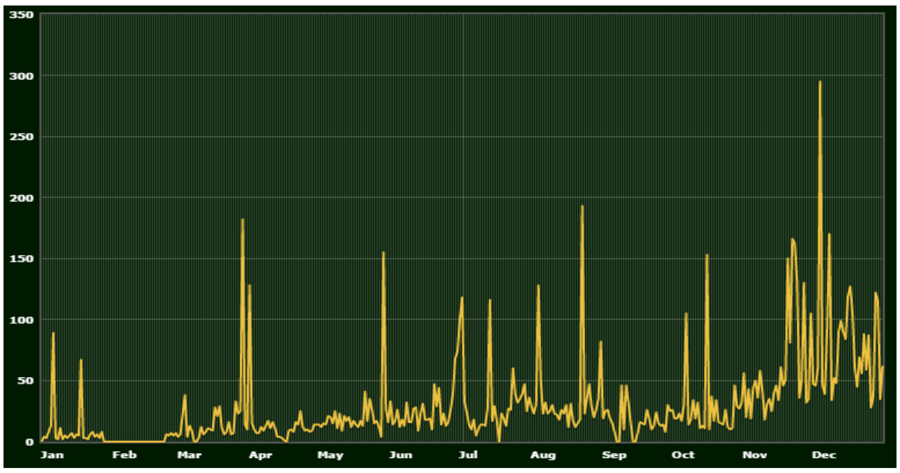
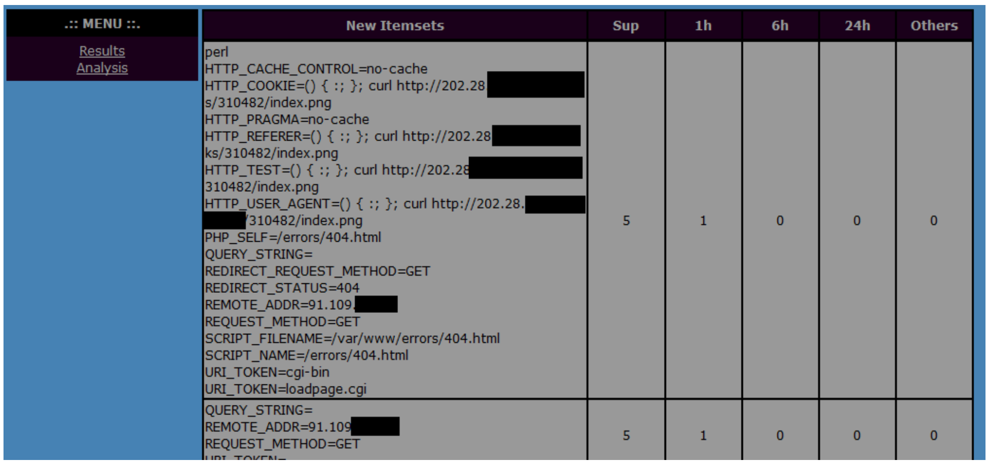


Figure 3. Activity observed by WebHP sensors during the year 2013. Chart generated by HPMS software

Additionally, tags can be created using information gained from other source, for example, articles, reports or personal knowledge. Moreover, data mining methods implemented in the HPMS system are very beneficial, too. For this purpose the miner module, which discovers frequent sets in the recorded data, is used. The miner module treats each transaction as a set, and each transactions' attributes as an item set. Detected frequent sets represent repeated activity. In many cases such pattern can be easily converted to the tag definition and used later for marking new connections. Fig. 4 shows a sample pattern detected automatically using data mining techniques. The presented pattern is associated with ShellShock vulnerability [7] in the Linux program `bash`. Despite that `bash` is used locally by a logged user in most cases, some applications that use CGI interface and `bash` can be attacked too.



New Itemsets	Sup	1h	6h	24h	Others
perl HTTP_CACHE_CONTROL=no-cache HTTP_COOKIE=() { :; }; curl http://202.28[redacted] s/310482/index.png HTTP_PRAGMA=no-cache HTTP_REFERER=() { :; }; curl http://202.28[redacted] ks/310482/index.png HTTP_TEST=() { :; }; curl http://202.28[redacted] 310482/index.png HTTP_USER_AGENT=() { :; }; curl http://202.28.[redacted] [redacted]/310482/index.png PHP_SELF=/errors/404.html QUERY_STRING= REDIRECT_REQUEST_METHOD=GET REDIRECT_STATUS=404 REMOTE_ADDR=91.109.[redacted] REQUEST_METHOD=GET SCRIPT_FILENAME=/var/www/errors/404.html SCRIPT_NAME=/errors/404.html URI_TOKEN=cgi-bin URI_TOKEN=loadpage.cgi	5	1	0	0	0
QUERY_STRING= REMOTE_ADDR=91.109[redacted] REQUEST_METHOD=GET	5	1	0	0	0

Figure 4. A frequent set, automatically detected by HPMS miner module, which is later used for tag definition

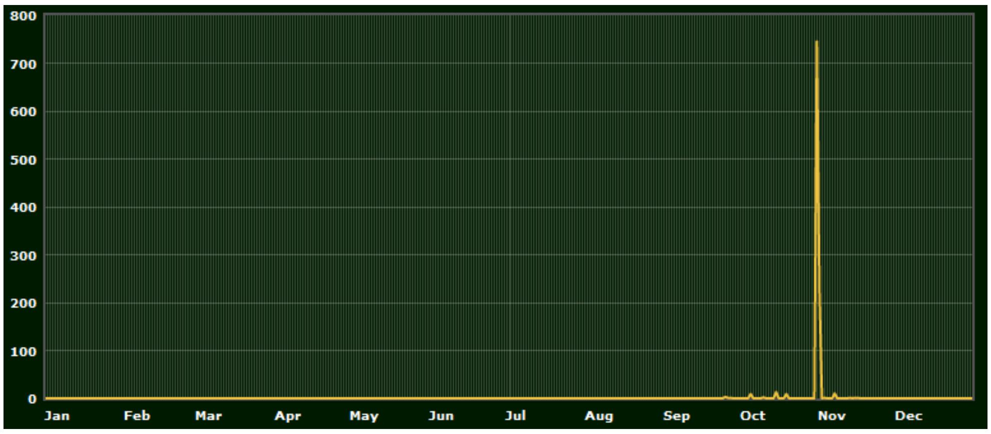


Figure 5. Transactions associated with ShellShock attack observed in 2014

When the tag is added, a new chart showing connections marked with recently added tag can be generated. In effect, after the detection of one attack, all similar ones can be detected too. The Fig. 5 shows connections marked with ShellShock tag, which was produced using knowledge gained from discovered frequent sets. In the figure, additional six attacks can be spotted besides the dominating one.

4. Analysis of PhpMyAdmin attack using graphs

As was described in the previous section sample charts can be very beneficial for data analysis. However, not all data can be easily visualized in this manner, sometimes more complicated methods must be used. As an example of advanced graphical methods which visualize attackers and victims are presented. Research of this particular attack starts when at the end of February 2014 a completely new activity appears in the deployed HoneyPot. Manual analysis shows that this attack uses vulnerability in the PhpMyAdmin [8], utility popularly used by system administrators for MySQL database management using web interface. Rapidly new offenders start using this exploit and more and more new malware samples are received. Fig. 6 shows an activity associated with this attack by the deployed WebHP sensors.

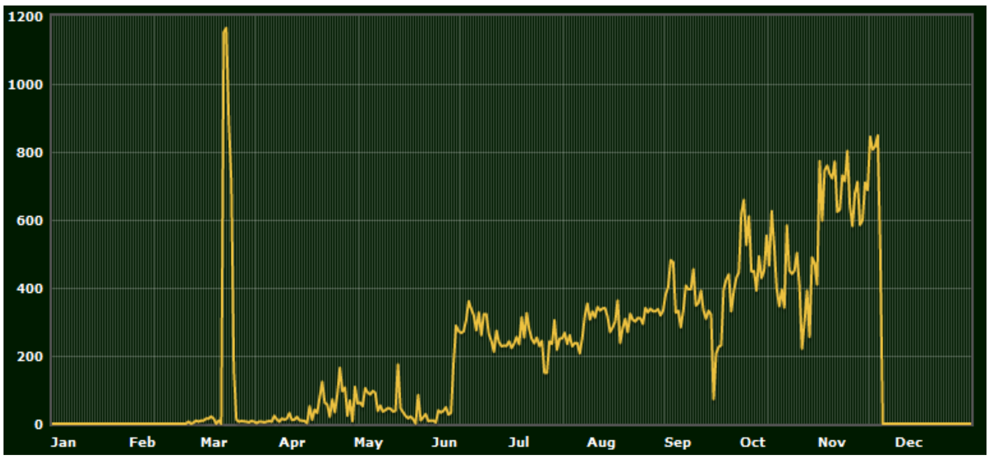


Figure 6. Automatically detected by HPMS miner module frequent set, which is later used for tag definition

Before the introduction of the visualization method, some more details concerning this attack must be presented. The first attacker’s request looks like an innocent attempt to gather the file /phpmyadmin/scripts/setup.php (see Fig. 7). The file name is a little strange, but the web server responds with appropriate data. What is important to the attacker, is that the returned file contains a unique session token which is later used during an injection attack.

```
GET /phpmyadmin/scripts/setup.php HTTP/1.1
Host: 194.29.XX.YY
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; MSIE 5.5; Windows NT
5.1) Opera 7.01 [en]

HTTP/1.1 200 OK
Date: Thu, 19 Jun 2014 08:51:35 GMT
Transfer-Encoding: chunked
Content-Type: text/html

24a7
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"><head>
. . .
     href="http:///app/phpMyAdmin/setup/index.php?version_check=1&a
mp;token=a60b82c06fa123c868288b29584d345">Check for latest version
</a>
. . .
```

Figure 7. Initial message and response during the attack to the PMA application gathered by WebHP system

The second message which is sent from the attacker to the victim contains an exploit. In this case the attacker utilizes code injection error in PMA application that allows remote change of configuration object. In effect the attacker can download and execute any malicious code on the victim machine. Message containing the exploit, which was sent to the HoneyPot is presented in the Fig. 8.

```
POST /phpmyadmin/scripts/setup.php HTTP/1.1
Host: 194.29.XX.YY
Referer: http://194.29.XX.YY/phpmyadmin/scripts/setup.php
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; MSIE 5.5; Windows NT
5.1) Opera 7.01 [en]
Content-Type: application/x-www-form-urlencoded
Content-Length: 238

action=lay_navigation&eoltype=unix&token=a60b82c06fa123c868288b29584
d345&configuration=a%3A1%3A%7Bi%3A0%3BO%3A10%3A%22PMA%5FConfig%22%3A
1%3A%7Bs%3A6%3A%22source%22%3Bs%3A29%3A%22ftp%3A%2F%2F37%2E59%2EAA%2
EBB%2Fpub%2F124%2Ephp%22%3B%7D%7D
```

Figure 8. Message containing an exploit used during the attack to the PMA application gathered by WebHP system

The first marked part of the message is gathered in the initial phase of the attack – the session token. The second marked part (after decoding) contains the following text: “a:1:{i:0;O:10:"PMA_Config":1:{s:6:"source";s:29: "ftp://37.59.AA. BB/pub/124.php";}}”

This text contains the specially crafted PMA software configuration object which could be remotely changed due to an error in the application. In effect, the vulnerable software injects, in the currently executed instructions, the code downloaded from the provided malicious URL – file 124.php from IP 37.59.AA.BB. Because this exploit is sent to the low interaction HoneyPot, this attempt was only stored in the data base and the malicious code did not execute. However, later manual analysis proved that the file 124.php contains malicious code which is used to remotely control the infected machine and to create a BotNet.

Sample analysis performed directly on data stored in the database could only find the most frequently used attackers, the used files or ftp servers from which malware was downloaded. This preliminary analysis rises interesting questions. If and how all those attackers, files and ftp servers are related? Is there any group of machines that works together? Do all such communities work is the same manner?

To answer all of these questions some visualization methods were introduced. Visualization utilizes simple undirected graphs, where a vertex represents an attacker (red), a filename (green) used by malware copy or an ftp server which hosts the malware (blue). The edge is added between a file and both an IP address – attacker and an ftp server. Conducted experiments prove the initial prediction that all the observed activity can be divided into independent groups. Fig. 9 shows a sample graph representing one of the detected groups.

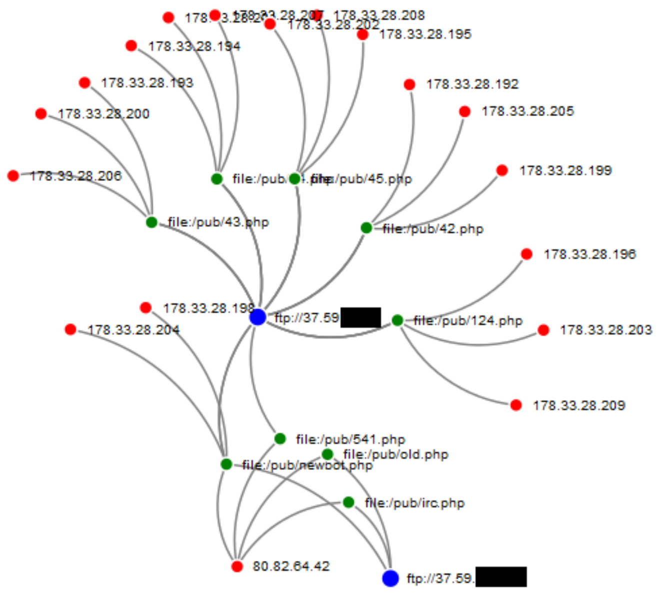


Figure 9. Graph presenting sample group of cooperating machines during attack

Graphs generated automatically by HPMS system rapidly presents groups of attacks that are joined by the used file name or the ftp server. However, many other interesting behaviors can be easily spotted in these pictures. Fig. 9 shows that most activity is associated with one ftp server which hosts many files. Additionally, each file in most cases is used by three attackers. This probably is effect of redundancy introduced by attackers. Fig. 10 shows three other groups of offending machines.

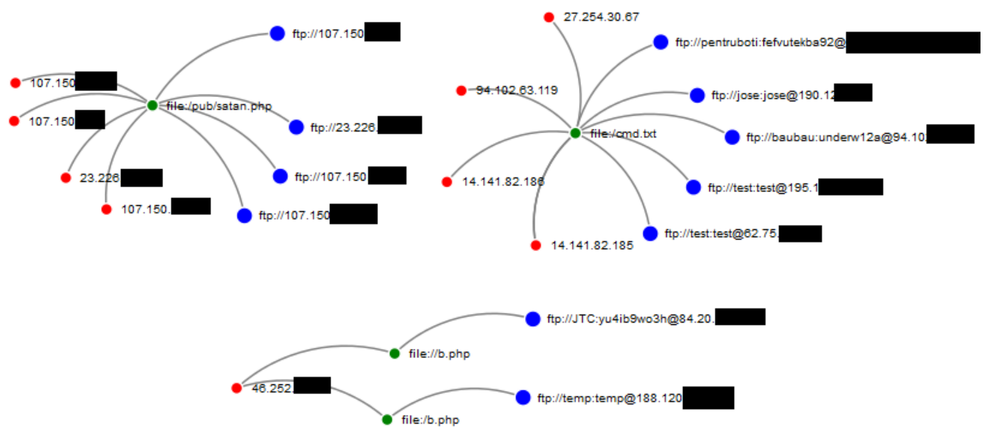


Figure 10. Graphs presenting sample groups of offending machines. Two top groups are file oriented. The third at the bottom is attacker oriented

The two graphs at the top of the Fig. 10 are file oriented, in contrast to the previous one presented in the Fig. 9 which is ftp server oriented. All attackers and ftp servers use the same file containing malware. What is interesting, in the graph at the left, all machines after infection download malware directly from the attacker IP. This is a sign that probably this is an automatic activity – some kind of the worm. The graph presented at the bottom of the Fig. 10 shows the attacker oriented graph. The central point of the graph is associated with an IP address of the offending machine. Used file has the same name, although lack of file name unification introduces two names – the only change is associated with the number of '/' in the file name, which in actual connection has no effect. Like the graph at the roght top of the figure 10, all the used ftp servers have login and password set. Used passwords are not so trivial which suggests that all servers are hacked, and store malware without owners knowledge and permission.

5. Conclusions

Deployment of the HoneyPot systems is just the beginning. Valuable knowledge can be gained by careful analysis of data which is gathered by the deployed HoneyPots. The real work with these systems begins at this point. However, currently most efforts concerning HoneyPot software development is associated with an introduction of new types of HoneyPots [9, 10] or hiding the existing ones [11]. The development of analytical software for HoneyPots is unfortunately neglected. The HPMS systems presented in this paper have been developed due to that no suitable analytical software exists. Huge amount of recorded data needs additional analytical functionalities. The visualization and filtering functionalities, initially implemented as support tools, rapidly confirmed their usefulness. As was presented in the previous paragraphs these simple methods can be very beneficial for people who perform analysis of data recorded by HoneyPot systems. The methodology described in the section 3, despite its simplicity, can be used with profound effects to analyze and characterize the activity observed by HoneyPot systems in the given time range.

Acknowledgments

Works associated with analysis of data obtained from HoneyPot system was funded by the Deans grant no. II/2014/GD/1 titled “Utilization of data mining techniques for network security data analysis”.

REFERENCES

[1] Cabaj K., Denis M., Buda M. (2013) Management and Analytical Software for Data Gathered from HoneyPot System, Information Systems in Management, WULS Press Warsaw, vol. 2, nr 3, 182-193

[2] Cheswick B. (1992) An Evening with Berferd in which a cracker is Lured, Endured, and Studied, In Proc. Winter USENIX Conference

[3] Provos N., Holz T. (2008) Virtual Honeypots: From Botnet Tracking to Intrusion Detection, Addison-Wesley

[4] Baecher P., Koetter M., Dornseif M., Freiling F. (2006), The nepenthes platform: An efficient approach to collect malware, In Proceedings of the 9 th International Symposium on Recent Advances in Intrusion Detection (RAID06)

[5] Dionaea catches bugs, http://dionaea.carnivore.it/ [2014.11.29]

[6] Cabaj K., Gawkowski P. (2014) HoneyPot systems in practice, The Nineteenth International Multi-Conference On Advanced Computer Systems (ACS14), Międzyzdroje, Poland, October 22 - 24

[7] Shell Shock attack, http://en.wikipedia.org/wiki/Shellshock_%28software_bug%29 [2014.11.29]

[8] PhpMyAdmin, www.phpmyadmin.net/ [2014.11.29]

[9] Bringer M. L., Chelmecki C. A., Fujinoki H., (2012) A Survey: Recent Advances and Future Trends in Honeypot Research, I. J. Computer Network and Information Security 10, 63-75

[10] N. Provos, T. Holz, Praise for virtual HoneyPots, Pearson Education, ISBN 978-0-321-33632-3, (2007)

[11] Fu X., Yu W., Cheng D., Tan X., Streff K., and Graham S., (2006) On Recognizing Virtual Honeypots and Countermeasures,‖ Proceedings of the IEEE International Symposium on Dependable, Autonomic and Secure Computing, pp. 211-218