

# A New Procedure to Help System/Network Administrators Identify Multiple Rootkit Infections

Desmond Lobo, Paul Watters and Xin-Wen Wu

Internet Commerce Security Laboratory

University of Ballarat

Ballarat, Australia

desmondlobo@students.ballarat.edu.au, {p.watters, x.wu}@ballarat.edu.au

**Abstract**— Rootkits refer to software that is used to hide the presence of malware from system/network administrators and permit an attacker to take control of a computer. In our previous work, we designed a system that would categorize rootkits based on the hooks that had been created. Focusing on rootkits that use inline function hooking techniques, we showed that our system could successfully categorize a sample of rootkits using unsupervised EM clustering.

In this paper, we extend our previous work by outlining a new procedure to help system/network administrators identify the rootkits that have infected their machines. Using a logistic regression model for profiling families of rootkits, we were able to identify at least one of the rootkits that had infected each of the systems that we tested.

**Keywords**- network security; malware; rootkits; logistic regression; profiling

## I. INTRODUCTION

When some malicious software (malware) manages to infiltrate a computer network, it can then spread throughout the network, infecting the computers connected to it. The writers of these types of software have moved away from creating malware for fun and have started developing code with the intention of huge financial gains. We are no longer dealing with script kiddies just trying to create havoc, but instead are targeted by organized criminal gangs that are deploying advanced forms of malware, with rootkit capabilities, on the Internet. It is estimated that 85% of malicious code is being written today with the goal of creating profit for the malware's author [1].

Rootkits refer to software that is used to hide the presence of malware (such as viruses, worms and trojans) from system/network administrators and permit an attacker to take control of a computer system [2]. In fact, installing a rootkit is usually the first thing that an attacker will do after gaining access to a system [3]. This will ensure that the attacker will remain undetected.

There exist many different types of rootkits. In this the initial phase of our research, we focus strictly on rootkits that use inline function hooking techniques [4], but plan to include other types of rootkits in the later stages of our work.

There are several tools available, such as McAfee's Rootkit Detective, which can be used to detect the inline function hooks that have been created by one or more

rootkits on a computer system. Even with this knowledge, however, it is still not possible to determine what rootkits have infected the machine. Each time a tool such as the Rootkit Detective is run, a log file is generated that contains a list of the detected hooks. The amount of data in these log files is overwhelming as they often contain thousands of lines of text, several lines for each detected hook. Figure 1 is an extract from one of these log files. This data represents just three of the many inline function hooks that had been created by a Berbew rootkit.

The main contribution of this paper is that we have devised a new procedure that will help system/network administrators make sense of the vast amount of data in these log files. This procedure will help these administrators identify the rootkits that have infected their computers. It is a two-step process and these steps are described in sections II and III. A discussion of some of the related work is provided in section IV. This is followed by an outline of our future work in section V and a conclusion to the paper in section VI.

```

McAfee(R) Rootkit Detective 1.1 scan report
On 16-08-2009 at 03:41:08
OS-Version 5.1.2600
Service Pack 3.0

=====

Object-Type: IAT/EAT-hook
PID: 2604
Details: Export : Function : kernel32.dll!Process32Next =>
C:\WINDOWS\system32\kernel32.dll:7C80029C
Object-Path: C:\WINDOWS\system32\kernel32.dll
Status: Hooked

Object-Type: IAT/EAT-hook
PID: 2412
Details: Export : Function : ADVAPI32.dll!RegEnumValueW =>
C:\WINDOWS\system32\ADVAPI32.dll:77DD04B4
Object-Path: C:\WINDOWS\system32\ADVAPI32.dll
Status: Hooked

Object-Type: IAT/EAT-hook
PID: 2252
Details: Export : Function : ntdll.dll!ZwQuerySystemInformation
=> C:\WINDOWS\system32\ntdll.dll:7C90027C
Object-Path: C:\WINDOWS\system32\ntdll.dll
Status: Hooked

```

Figure 1. Inline Function Hooks

TABLE I. RESULTS FROM EXPERIMENT ONE

Antivirus Labels	Family	Number of Samples
ClamAV Worm.Korgo.Z BitDefender Backdoor.Berbew.Be.DAM	F <sub>1</sub>	21
ClamAV Trojan.Crypted-29 FProt W32.Berbew.F	F <sub>2</sub>	13
ClamAV Trojan.Crypted-29 FProt W32.Berbew.G	F <sub>2</sub>	11
ClamAV Trojan.Qukart-8 FProt W32.Berbew.F	F <sub>2</sub>	1
ClamAV Trojan.Qukart-10 FProt W32.Berbew.F	F <sub>2</sub>	1
ClamAV Trojan.Qukart-17 FProt W32.Berbew.F	F <sub>2</sub>	1
ClamAV Trojan.Crypted-28 AVGScan I-Worm.Nuwar.N FProt W32.Berbew.F	F <sub>2</sub>	6
ClamAV Trojan.Crypted-28 BitDefender Trojan.Spy.Qukart.Z FProt W32.Berbew.G	F <sub>2</sub>	2
ClamAV Trojan.Bancos-6278	F <sub>3</sub>	2
ClamAV Worm.Feebs.AE BitDefender Win32.Worm.Feebs.I.Gen	F <sub>4</sub>	14
Kaspersky Worm.Win32.Feebs.a	F <sub>4</sub>	1
BitDefender Trojan.PWS.Papras.A	F <sub>5</sub>	1
BitDefender Trojan.PWS.Papras.O	F <sub>5</sub>	1
BitDefender Trojan.PWS.Papras.F	F <sub>5</sub>	1
BitDefender Trojan.PWS.Papras.L	F <sub>5</sub>	2

## II. OUR PREVIOUS WORK

The procedure to identify the rootkits that have infected a machine is a two step process. In our previous work [5], we described the first step, our first experiment, which involved the categorizing of a sample of rootkits into several groups based on the inline function hooks that had been created by each of these rootkits.

Having obtained and tested 78 rootkit samples, we found that a total of 11,159 inline function hooks had been created. Based on the particular hooks that had been created by each rootkit and using the unsupervised EM clustering algorithm, we were then able to categorize the samples into groups.

The clustering algorithm calculated the optimum number of clusters to be five and generated the following results. The 78 rootkit samples were clustered into these five groups:

- There were a total of 21 samples in the first group, which will subsequently be referred to as family F<sub>1</sub>.
- There were a total of 35 samples in the second group, which will subsequently be referred to as family F<sub>2</sub>.
- There were a total of 2 samples in the third group, which will subsequently be referred to as family F<sub>3</sub>.
- There were a total of 15 samples in the fourth group, which will subsequently be referred to as family F<sub>4</sub>.
- There were a total of 5 samples in the fifth group, which will subsequently be referred to as family F<sub>5</sub>.

The next thing to do was to validate these results: had our system generated meaningful clusters? To answer this question, we proceeded by labeling the rootkit samples using several different antivirus scanners.

The rootkit samples for our experiment had all been obtained from the Offensive Computing website (<http://www.offensivecomputing.net>). This website uses the following five antivirus scanners to label their malware samples:

FProt  
BitDefender  
Kaspersky  
ClamAV  
AVGScan

The 78 samples that we had obtained from this website had each been labeled by at least one of these antivirus scanners.

- The 21 samples in family F<sub>1</sub> had something in common: they had all been labeled as Berbew by the BitDefender antivirus scanner.
- The 35 samples in family F<sub>2</sub> had something in common: they had all been labeled as Berbew by the FProt antivirus scanner.
- The 2 samples in family F<sub>3</sub> had something in common: they had both been labeled as Bancos.
- The 15 samples in family F<sub>4</sub> had something in common: they had all been labeled as Feebs.
- The 5 samples in family F<sub>5</sub> had something in common: they had all been labeled as Papras.

Given that the samples within each family had something in common, we concluded that the results that had been generated by the EM clustering algorithm, which are summarized in Table I, were very reasonable. In essence, we showed that our system could be used to successfully categorize a sample of rootkits that use inline function hooking techniques.

## III. TESTING MULTIPLE ROOTKIT INFECTIONS

When conducting our first experiment, we had tested just one rootkit sample at a time on a clean system, and then restored the system back to a clean state before testing the next sample. In reality, though, it is often the case that a machine is found with more than one infection. Thus, in our second experiment, the second step in the process, we wanted to determine if our identification technique could handle the situation in which more than one rootkit was running on the system.

We proceeded as follows. The dataset for our first experiment had consisted of 78 instances (one instance for each rootkit) and 1560 attributes for each instance. The 1560 attributes had contained binary values and represented the 1560 hooks that we had tested, with a positive value for a certain attribute meaning that that particular hook was present. As mentioned, we had managed to cluster the 78 rootkits into five families using these attributes.

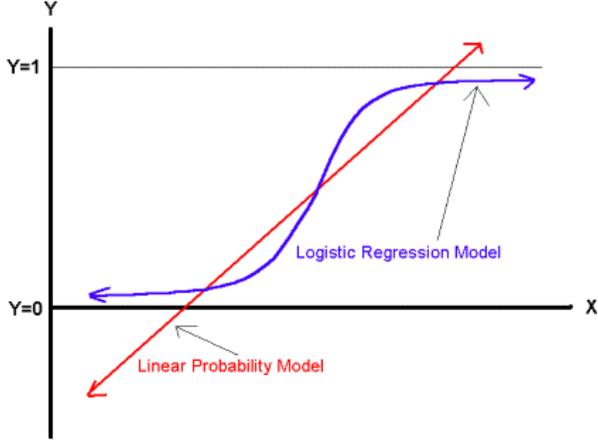


Figure 2. Linear Probability Versus Logistic Regression [6]

For our second experiment, we added one more attribute to our dataset. This 1561<sup>st</sup> attribute, which we will call a marker, contained the name of the family that the rootkit belonged to.

#### A. Logistic Regression

Having a marker for each of the 78 instances, we then made use of a logistic regression model for profiling the families of rootkits. We used this particular model since it can be used in data (where the family is known) to find similarities between the instances within each family [7]. In other words, we used this model to find the attributes that could be used to differentiate between the different families of rootkits.

In a linear probability model, the data is fitted to a straight line, but this type of model is not always appropriate. For logistic regression, on the other hand, an S-shaped curve is used instead. This is illustrated in Figure 2. An advantage of such an S-shaped curve is that it gives you a value between 0 and 1, and this can be interpreted as the probability of belonging to a certain family [8]. Cutoff values are established to determine if an instance belongs to a particular family [7].

The probability function that is used to determine the acceptance of an instance  $i$  to a family  $j$  is given by the formula

$$P_j = \frac{1}{1 + e^{(-\beta_0 - \sum \beta_i X_i)}} \quad (1)$$

where the  $\beta$  coefficients are calculated using Maximum Likelihood Estimation (MLE) [9]. The  $X_i$  variables contain binary 0/1 values and represent the attributes for each of the instances.

#### B. Results

The logistic regression model was available for implementation in the Wakaito Environment for Knowledge Analysis (WEKA) [10]. The results after applying the model are illustrated in Figure 3. We found that:

- Attribute 313 could be used to identify a rootkit from family  $F_1$
- Attribute 1423 could be used to identify a rootkit from family  $F_2$
- Attribute 4 could be used to identify a rootkit from family  $F_3$
- Attribute 970 could be used to identify a rootkit from family  $F_4$
- Attribute 24 could be used to identify a rootkit from family  $F_5$

Having many attributes and just a handful of families, it was not surprising that we could identify each family using just one particular attribute. Thus, in this case, it was fairly straightforward; this same logistic regression model could, however, even be used in more complex circumstances.

Next, we tested our system's ability to detect multiple rootkit infections by trying many different combinations of rootkits. The results of these tests are summarized in Table II and a brief explanation of these results follows.

The second row of the table displays the results of a test in which the machine was infected with two rootkits, one from family  $F_4$  and then one from family  $F_5$ , and our system managed to identify both families. On the other hand, the fourth row of the table displays the results of a test in which the machine was again infected with two rootkits, this time one from family  $F_1$  and then one from family  $F_2$ , but our system only managed to identify the rootkit from family  $F_1$ .

It should be pointed out that we used identical rootkit samples for each test within a group. So, for example, for group 1, we first picked a rootkit from family  $F_4$  and then one from family  $F_5$ . For the following test, we used the same rootkits but just reversed the order.

It should also be noted that when testing more than one rootkit on our system, we found that certain combinations of rootkits caused our system to crash.

Based on our experimental results in Table II, we drew the following conclusions:

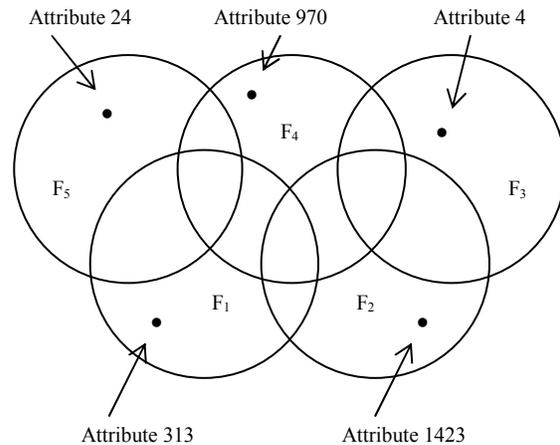


Figure 3. Attributes that Identify each Family

TABLE II. RESULTS FROM EXPERIMENT TWO

Group	1 <sup>st</sup>	2 <sup>nd</sup>	3 <sup>rd</sup>	Attr. 313	Attr. 1423	Attr. 4	Attr. 970	Attr. 24
1	F <sub>4</sub>	F <sub>5</sub>	-	×	×	×	✓	✓
1	F <sub>5</sub>	F <sub>4</sub>	-	×	×	×	✓	✓
2	F <sub>1</sub>	F <sub>2</sub>	-	✓	×	×	×	×
2	F <sub>2</sub>	F <sub>1</sub>	-	✓	×	×	×	×
3	F <sub>3</sub>	F <sub>4</sub>	-	×	×	✓	✓	×
3	F <sub>4</sub>	F <sub>3</sub>	-	×	×	✓	✓	×
4	F <sub>1</sub>	F <sub>5</sub>	-	✓	×	×	×	✓
4	F <sub>5</sub>	F <sub>1</sub>	-	✓	×	×	×	✓
5	F <sub>2</sub>	F <sub>5</sub>	-	×	×	×	×	✓
5	F <sub>5</sub>	F <sub>2</sub>	-	×	×	×	×	✓
6	F <sub>1</sub>	F <sub>4</sub>	-	✓	×	×	✓	×
6	F <sub>4</sub>	F <sub>1</sub>	-	✓	×	×	✓	×
7	F <sub>2</sub>	F <sub>4</sub>	-	×	✓	×	✓	×
7	F <sub>4</sub>	F <sub>2</sub>	-	×	✓	×	✓	×
8	F <sub>2</sub>	F <sub>3</sub>	-	×	✓	×	×	×
8	F <sub>3</sub>	F <sub>2</sub>	-	×	✓	✓	×	×
9	F <sub>4</sub>	F <sub>1</sub>	F <sub>5</sub>	✓	×	×	✓	✓
9	F <sub>5</sub>	F <sub>4</sub>	F <sub>1</sub>	✓	×	×	✓	✓
9	F <sub>1</sub>	F <sub>5</sub>	F <sub>4</sub>	✓	×	×	✓	✓
10	F <sub>2</sub>	F <sub>3</sub>	F <sub>1</sub>	✓	×	×	×	×
10	F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>	✓	×	×	×	×
10	F <sub>3</sub>	F <sub>1</sub>	F <sub>2</sub>	✓	×	×	×	×
11	F <sub>1</sub>	F <sub>2</sub>	F <sub>4</sub>	✓	×	×	✓	×
11	F <sub>4</sub>	F <sub>2</sub>	F <sub>1</sub>	✓	×	×	✓	×
11	F <sub>2</sub>	F <sub>4</sub>	F <sub>1</sub>	✓	×	×	✓	×

1. When we had a system with more than one rootkit infection, we were able to identify the family of at least one of the rootkits that had infected the system.
2. The order that the rootkits were executed does, in some cases, make a difference. After observing the results of testing rootkits from group 8, it seems that the rootkit that is executed first does appear to get the upper hand.

We hope that system/network administrators might benefit from this system. If an administrator detects some inline function hooks on a particular network machine, he or she could then use our identification procedure to determine the family of at least one of the rootkits that had created the hooks. Armed with that knowledge, the administrator could then proceed and take further action.

#### IV. RELATED WORK

Kolter et al. [11] extracted n-grams from malware samples and then tried to classify the samples into families. Like our work, they also made use of the WEKA data mining software package to analyze their datasets. Their approach, however, involved a static analysis of the malicious executable files, while our strategy was to dynamically analyze a live system to identify the rootkit infections.

Bayer et al. [12] collected malware samples and then performed a dynamic analysis of each sample in a controlled environment, eventually ending up with clusters of families. The objective of their research was to cluster malware samples based on the behavior of each sample. Bailey et al. [13] also carried out dynamic analyses of malware samples by executing each sample in a virtualized environment. They too categorized the malware samples into groups based on the malware's behavior. Our approach was similar to that of Bayer et al. and Bailey et al., but more focused. Instead of analyzing all types of malware, we concentrated exclusively on rootkits and used the inline function hooks that had been created by these rootkits to differentiate between the various families.

There has been some previous research that also used data mining techniques for profiling. Fawcett and Provost [14] designed an automatic method to detect fraudulent usage of cellular telephones, and this was based on the profiling of customer behavior using rule-based data mining methods. They first combed through their data in search of indicators of fraud. Rules were then established that could distinguish fraudulent calls from legitimate calls.

Huntington et al. [15] attempted to distinguish between the various groups of users that sought information about health on the Web. They profiled users based on the health sites that they visited. A logistic regression model was used to determine which site attributes could be used to identify the different user groups.

The work of Pepyne et al. [16] is most closely related to ours in the sense that they too applied logistic regression profiling techniques to the area of network security. Pepyne et al. attempted to profile computer users based on the temporal aspects of their behavior. If an individual's behavior deviated significantly from his or her group's profile, then there would be grounds for further inspection.

For our work, we too made use of a logistic regression model for profiling. Our strategy was unique, however, in that we devised a new procedure to use the profiles of rootkit families to identify the rootkits on a live system.

#### V. FUTURE WORK

This research focused strictly on rootkits that use inline function hooking techniques. As mentioned earlier, we plan to include other types of rootkits in the next stage of our work; in particular, we plan to analyze rootkits that hook the import address tables (IATs), export address tables (EATs) and system service descriptor tables (SSDTs).

Having a large and sparse dataset, we also plan to look into the possibility of using techniques from principal components analysis (PCA) for our study.

## VI. CONCLUSION

When malware manages to penetrate a computer network and jump from machine to machine, it can create a lot of difficulty for the system/network administrator. The emergence and proliferation of rootkits, which are used to hide the presence and activity of malware, are only going to make this problem worse.

In this paper, we focused on rootkits that use inline function hooking techniques to remain hidden. There are several tools available, such as McAfee's Rootkit Detective, that can be used to find the hooks that have been created on a system, but these tools cannot be used to identify the rootkits.

The most important contribution of this paper is that we have developed a new procedure that will help system/network administrators identify the rootkits that have infected their machines. We used a logistic regression model for profiling the rootkit families and then determined which attributes could be used to differentiate between the various families.

After testing several systems that had been infected with more than one rootkit, we found that we could identify at least one of the rootkit families. This knowledge would certainly help the administrator in deciding how to proceed.

## ACKNOWLEDGMENT

This research was supported in part by the Westpac Banking Corporation, IBM Australia and the Victorian Government in Australia.

We would like to thank Prof Lynn Batten for her helpful comments.

## REFERENCES

- [1] L. Wang and P. Dasgupta, "Kernel and Application Integrity Assurance: Ensuring Freedom from Rootkits and Malware in a Computer System", Proceedings of the 21<sup>st</sup> International Conference on Advanced Information Networking and Applications Workshops, 2007, IEEE Computer Society
- [2] A. Emigh, "The Crimeware Landscape: Malware, Phishing, Identity Theft and Beyond", Journal of Digital Forensic Practice, Vol. 1(3), Sept. 2006, pp. 245-260
- [3] M. Alvarez, M. Vucelich, and L. Johnson, "IBM Internet Security Systems X-Force Threat Insight Monthly", July 2008, IBM Corporation
- [4] G. Hoglund, and J. Butler, "Rootkits: Subverting the Windows Kernel", 2005, Addison-Wesley Professional
- [5] D. Lobo, P. Watters and X. Wu, (in press) "RBACS: Rootkit Behavioral Analysis and Classification System", Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, 2010, IEEE Computer Society
- [6] J. C. Whitehead, "An Introduction to Logistic Regression", Retrieved October 8, 2009 from <http://www.appstate.edu/~whiteheadjc/>
- [7] D. Olson and Y. Shi, "Introduction to Business Data Mining", 2007, McGraw-Hill Irwin
- [8] M. H. Dunham, "Data Mining: Introductory and Advanced Topics", 2003, Pearson Education
- [9] G. Shmueli, N. R. Patel, and P. C. Bruce, "Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner", 2007, John Wiley & Sons
- [10] I. H. Witten and E. Frank, "Data Mining: Practical machine learning tools and techniques", 2nd Edition, 2005, Morgan Kaufmann
- [11] J. Z. Kolter, "Learning to Detect and Classify Malicious Executables in the Wild", The Journal of Machine Learning Research, Vol. 7, Dec. 2006, pp. 2721-2744
- [12] U. Bayer, P. Milani Comparetti, C. Hlauschek, C. Kruegel and E. Kirda, "Scalable, Behavior-Based Malware Clustering", Proceedings of the 16th Annual Network & Distributed System Security Symposium, 2009
- [13] M. Bailey, J. Oberheide, J. Andersen, Z. M. Mao, F. Jahanian and J. Nazario, 2007, "Automated Classification and Analysis of Internet Malware", Recent Advances in Intrusion Detection, Lecture Notes in Computer Science, Vol. 4637, pp. 178-197, Springer
- [14] T. Fawcett and F. Provost, "Combining Data Mining and Machine Learning for Effective User Profiling", Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996, Association for the Advancement of Artificial Intelligence
- [15] P. Huntington, D. Nicholas, and P. Williams, "Characterising and profiling health Web user and site types: going beyond hits", Aslib Proceedings, Vol. 55, Issue 5/6, 2003, pp. 277-289
- [16] D. L. Pepyne, J. Hu, and W. Gong, "User Profiling for Computer Security", Proceeding of the American Control Conference, Vol. 2, 2004, pp. 982-987, IEEE Xplore