A Novel Methodology for Malware Intrusion Attack Path Reconstruction

Ahmed F. Shosha, Joshua I. James, and Pavel Gladyshev

School of Computer Science and Informatics
Centre for Cyber Security and Cyber Crime Investigation
University College Dublin, Ireland
ahmed.shosha@ucdconnect.ie,
{joshua.james, pavel.gladychev}@ucd.ie

Abstract. After an intrusion has propagated between hosts, or even between networks, determining the propagation path is critical to assess exploited network vulnerabilities, and also to determine the vector and intent of the initial intrusion. This work proposes a novel method for malware intrusion attack path reconstruction that extends post-mortem system state comparison methods with network-level correlation and timeline analysis. This work shows that intrusion-related events can be reconstructed at the host level and correlated between related hosts and networks to reconstruct the overall path of an attack. A case study is given that demonstrates the applicability of the attack path reconstruction technique.

Keywords: Malware Analysis, Attack Path Reconstruction, Digital Forensics, Network Forensics, Automatic Event Reconstruction

1 Introduction

The Internet has become an essential part of the daily life of many people, as well as the backbone of almost all business, services and industries. However, the Internet not only brings a wealth of commerce, communication and knowledge sharing, but also new threats in the form of malware, digital attacks and privacy leaks.

Over the past few years, antivirus vendors have shown that malicious programs that are developed for illegal purposes have grown dramatically. Symantec security threats report recorded over 3 billion malware attacks in 2010, with a 93% increase in web based attacks compared to 2009 [1]. The attacks include targeted attacks on critical infrastructure networks such as Supervisory Control And Data Acquisition (SCADA) networks attacks (a.k.a Stuxnet malware), the Hydraq attack (a.k.a Aroura), Social networking malware intrusions, and mobile malware intrusions. Much of this has been developed to support professional computer criminals and Internet gangs. This overwhelming increase use of the malware intrusions by computer criminals combined with advanced techniques to hide the presence of malicious code, complicates the digital investigation process, and disrupts traditional digital investigation techniques. Thus, a novel, highly automated forensics analysis technique

to allow digital investigators to cope with a growing number of malware threats and discover traces of intrusions is required.

One of the most important attack vectors used by malware intrusions are network vulnerabilities. Malware intrusions exploit network vulnerabilities to propagate over hosts, and from one network to another, which makes the process of determining the source of the attack is very difficult.

In this paper, a novel methodology for post-mortem analysis of hosts infected by malware is proposed in order to discover the propagation path of a malware attack. The post-mortem analysis of infected hosts is based on the detection of malware traces using event time bounding methods described by Gladyshev and Patel in [2]. Time bounding is used to derive an overall picture of host infection throughout the network. Malware infection times are determined based on the detection of malware traces in host systems using comparative methods described by Zhu, et al. [3] to reconstruct past digital events. This method allows the use of system snapshot data – for example, from Microsoft Windows System Restore Points [4] – to derive more information about past events than could be found with single-state analysis. After event information has been extracted, event time information, and specifically the initial time of infection, is correlated to derive the attack path history of the malware intrusion, as well as the intrusion source host or network of the attack. Consequently, the output of the methodology is a graph stating the malware intrusion attack path and propagation flow, which is used to infer the source host or network of the malware intrusion.

1.1 Contribution

The contribution of this paper is highlighted as follows:

A. A method is proposed for the identification of malware intrusion infection times based on past system states derived from Windows System Restore points.
B. A method to correlate multi-host malware infection times is used to infer the malware propagation path.
C. Timeline analysis of malware intrusion and malware propagation that results determination of the malware entry point.

1.2 Paper Organization

The paper is organized as follows: Section two gives a brief description of Microsoft Windows System Restore points and registry snapshots. Section three describes the malware infection time inference process using event time bounding, and the process of using the timeline correlation of the infected hosts to derive the malware intrusion attack path and propagation flow from one host to another or from one network to another. Section four provides a practical case study on Microsoft Windows systems, and how to use system restore points to infer malware intrusion infection times. Finally, section five concludes the paper and examines future work.

2 Windows Systems Restore Points

Windows system restore points are a valuable source of previous system state information in Windows systems, as described in [3] [5-8]. The Windows system restore process [9] monitors the operating system changes and saves the operating system's state (a current copy of Windows registry hives and other monitored files) as a restore point in a consecutive set of system states. If a system later has issues, the user can roll back the system to a previous stable state using the data from a restore point.
The Windows restore point backup process is triggered by the Windows operating system Kernel under the following circumstances:

- Every 24 hours
- When installing unsigned drivers
- When installing systems updates using automatic updates
- When start restoration point process
- Before program installation
- Manual creation a of restore point by the system user.

As described, the system restore process was designed to help the operating system recover from unpredictable events that may cause disruption of the operating system while performing critical processes such as installation of software programs or installation of unsigned device driver, etc.

Windows restore point state contains different items; most notably registry hives, but also other files, such as executable files and library files – also known as DLLs. These files are copied to the restore point folder upon creation, and are renamed with a unique name. A log file called change.log, contains both original name and the descriptor name for each file in the restore point. The files copied into the restore point snapshot are identified by an xml directive file located in `%System32%\Restore\Filelist.xml`, which specifies the files being monitored by the operating system, and what files are included in the restore point backup process [10]. The mentioned files include executable files, DLLs, Windows device driver's files, and other extensions. In addition, a snapshot of Windows registry hives containing the current configuration of the Windows operating system and the users' settings are copied into the restore point folder. The monitored files' metadata – including last modification, last accessed and last created times, known as MAC times – are not altered during the restore point creation process, which allows for a more accurate creation of a timeline for objects included in restore point folders.

Analysis of registry snapshots over consecutive restore point states can be a valuable source for tracing malware configurations in infected hosts. Registry snapshots comparison can show system changes, installation of new hardware and/or software, altering of current software settings, installing new services, changes in a program's configuration over time, network configurations changes, and changes to firewall and security settings. Digital investigators could infer a considerable amount of information by comparing various registry snapshots over certain time-span to

determine when a malware intrusion first infected the system, and what changes and configurations were the results of such an infection, as shown in [3] and [6].
Generally, installation of malware executables in the infected host also copies the malicious executable into the restore point folders, as most of malware code is developed in PE formats [11] such as executables and Windows drivers that are registered in the xml monitoring file (`Filelist.xml`). In addition, the majority of malware places their configuration in the Windows registry, which makes the process of registry examination inevitable.
Further, malware employs different techniques to maintain persistence, such as installing the malicious code as a system service or device driver, loading the malicious code at startup, and sometimes adding the settings and configuration of the malware to the registry hives. An example of the keys used by the malware include `<run>`, `<run once>`, `<Service>`, `<Browser Helper Objects>` and `<Internet Settings>` keys. This ensures persistence of the malicious code after the host has powered down.

3 Malware Infection Timeline Analysis

Forensic examination of a collection of saved states presented in Windows restore points is an important part of analyzing malware. The restore point investigation provides more information pertaining to the malware's activities in the infected host. This section describes a method for timeline analysis of an infected host using restore point information. The proposed method compares consecutive registry snapshot data to extract traces related to the infection, and to determine when the initial point of infection occurred.
When presented with a collection of Windows restore points, it is possible to identify a time-span in which the host was initially infected. When the malware is executed, traces of the execution may be created in the Windows registry. After the initial infection, when a restore point is created, a copy of the registry hive containing the traces of malware will be copied into the restore point folder. At a minimum, evidence of the malware execution process may be found within the Windows registry's `ShellBag` keys [12].
By examining past registry snapshots and determining when the malware-related traces first appeared, the host infection time can be determined. The resulting infection time will be used later to assist in the process of determining the malware intrusion infection path, how the malware propagated across the network, and the source of the malware intrusion.

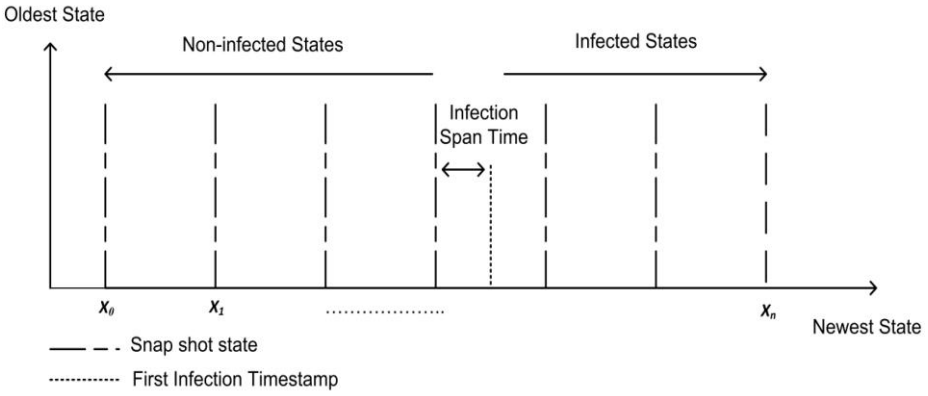

Figure 1: Bounding initial infection times within an infected host based on the appearance of malware traces across consecutive snapshots

The proposed infection time identification process is mainly dependent on two main procedures. The first procedure targeted the malware trace evidence present in registry keys, while the second procedure focuses on the comparative method of the evidence timestamps to identify the first time the malware was introduced to the infected system (initial infection time).

As shown in figure 1, the initial infection time is based on comparing the malware trace timestamps in consecutive registry snapshots to discover the first infection timestamp. The malware traces' timestamps are bound by two times denoted as T_{min} and T_{max} such that T_{min} describes the time when investigation process was started and T_{max} describes the first observed timestamp of malware traces. The malware infection period is defined as: $\Delta T = T_{max} - T_{min}$.

The infection period is determined by iterating over each observed malware trace E_m in every restore point. The procedure compares the timestamp of E_m in both restore points RP_x and RP_{x-1} until no trace timestamp is discovered. The last discovered E_m is then the initial time of infection.

One limitation with this method is single last-modified timestamp associated with Windows Registry keys. Malware trace evidence timestamps, such as malware service load times in the service configuration keys, will only include the last modified time for malware-related keys in the registry. Hence, if the malware re-loaded the service many times, the only timestamp that is available will be the most recent execution timestamp.

To overcome the aforementioned limitation, a time slack period, as defined in [12], is created from the first observed malware trace timestamp to the restore point creation timestamp to avoid multi-access problem of the evidence keys. In practice, the time slack that denoted as infection span time will be limited to a maximum 24 hours in the worst-case scenario.

3.1 Correlation of Infection Times

The malware intrusion attack path is a graph-based representation for paths that are likely to be used by malware to infect other hosts or networks. From a security prospective, malware intrusion attack path construction allows security analysts to assess the vulnerabilities of connected hosts and networks, as described in [13-15], and also to understand how vulnerabilities in an individual host can contribute to overall network vulnerability. In general, an attack graph's function is defensive since it used to assess the deployment scheme of Intrusion Detection Systems (IDS) and firewalls. In digital investigation, constructing malware attack paths enables digital investigators to identify the sources of malware-related attacks and attack activities in hosts and the network.

The proposed method for constructing malware intrusion attack paths is based on the correlation of the infection times derived from various infected hosts. A relationship between hosts is mandatory for the malware to propagate. The host relationships are presented in a connectivity meta-table. The hosts are connected together in the network based on the design of network topology. The connectivity meta-table is a representation of how each host is reachable from each other host, and over which ports and services. If the connectivity preconditions between the infected and non-infected hosts are satisfied, the infection probability factor is set to one; otherwise it is set to zero.

Correlating host infection times is a process to determine the probability that host A has infected host B based on infection times of both *A* and *B*, while considering the connectivity constrains between *A* and *B*.

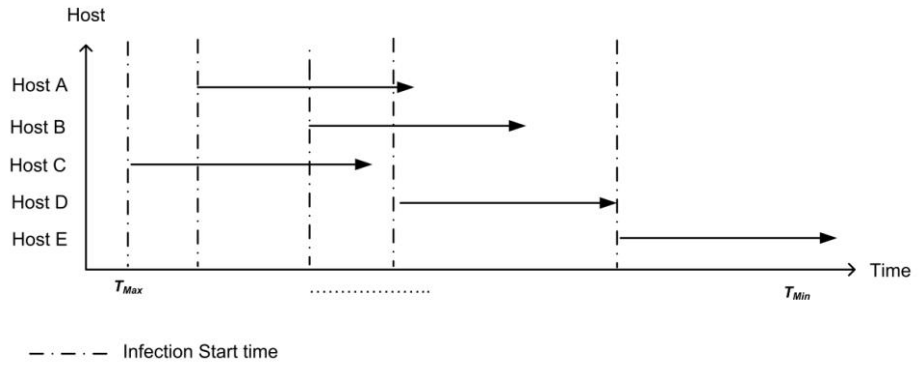

Figure 2: Identified possible host infection time-spans for 5 related hosts

As shown in figure 2, the probability that a certain host, *X,* is infected from another host is governed by T_{max} of *X* and T_{max} of the other host. Thus, if $T_{max}(x) > T_{max}(x+1)$ then, $P\ (T_{max}(x))$ infected from $T_{max}(x+1)$ = one otherwise zero where value one is most likely to be infected and zero most likely is not infected.

The process of correlating the host infection times to other hosts after satisfying the connectivity conditions and constrains is continued for all hosts. The process results

in a graph stating the infection propagation path from one host to another, as well as the entry point of the malware. Re-running the process over multiple networks will clearly show how malware has propagated from one network to another, and what network is the source of attack.

4 Case Study

To demonstrate how the aforementioned infection time bounding methodology is applied, a practical case study of a malicious code infection is presented. The case involves dynamic analysis malware propagation over multiple hosts and networks. A preliminary setup for malware test environments includes installation of nine Windows XP hosts in a virtual machines environment. The virtual machines are separated in three networks, and connected via a simulated routing service. 21 test cases of malicious worm infections are executed by infecting a machine from network A and observing the malware propagation over network A hosts, then from network A to both network B and C. The malware samples collected from many sources include malware research sites and honeynets [16] [17]. In each virtual machine, a clean installation of Microsoft Windows XP SP3 was installed and operated in normal user mode for five days. A number of 7 to 11 restore points were created in first five days in each virtual machine due to different software application installations, and a default restore point creation every 24 hours. Malware was introduced into the first virtual machine; after, the infected host was left for three days to allow the malware to propagate to other hosts and networks. Three days after the first host was infected, forensic images of all infected virtual machines were taken using Access Data's Forensic Tool Kit (FTK) software [18]. A Python script was developed to extract restore point folders and examine the extracted registry hive snapshots for malware evidences traces. The developed script outputs a timeline of malware propagation based on the information extracted from the registry snapshots. An xml file, containing a list of keys commonly used by malware, was used to direct the malware trace search process. The keys include, but are not limited to, auto run keys, services keys, ShellBag keys, network connection and network setting keys, Internet history, browser helper objects and firewall settings.

The search process is designed to search for traces of particular malware in the specified keys within all restore points. If traces are found, the trace timestamps are compared to other restore points to identify the first time the system was infected with the malware. Once the operation is complete for all traces in the infected host, an xml file with the host's infection time information is generated. The resulting infection timeline for each host is shown in Figure 3 and 4.

After generation of the infection time information for all hosts, a different Python script correlates the information presented in the xml files, and produces a timeline of the malware propagation path, showing how and when the malware propagated from one network to another and the most likely source of the intrusion.

As shown in Figure 3, the timeline of the malware intrusion propagation and the attack path, clearly show that the malware intrusion initially started from Host one, then propagated to the other hosts within the three days. Malware propagation between different networks will only be from the hosts that have connectivity

privileges between networks. As shown in Figure 4, only the infection source hosts have access to more than one network, thus, infection in the new network was found to propagate from these hosts.

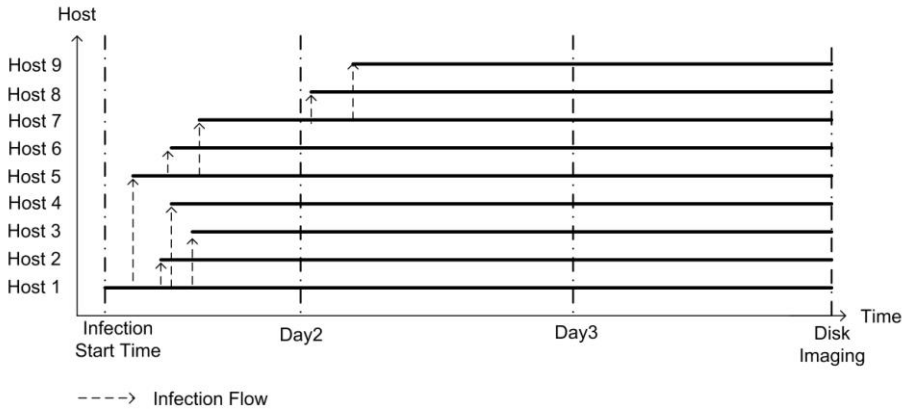

Figure 3: Malware infection and propagation times in nine hosts based on the detection of common malware traces within the Windows registry

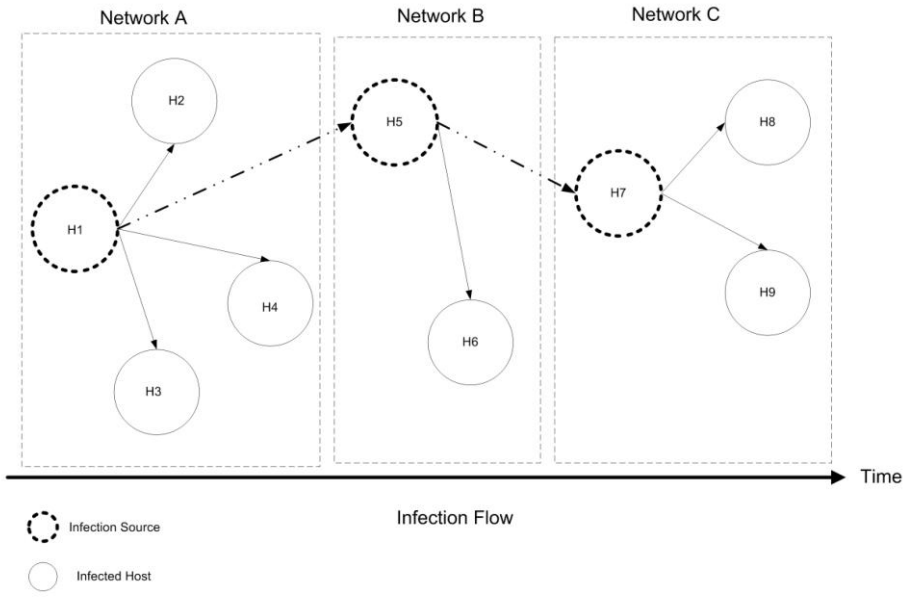

Figure 4: Malicious code propagation determined by correlation of initial host infection times across three networks where the border devices were found to be the infection sources

Digital investigation of the host event logs show the network connections used by the malware intrusion to connect to other hosts, forming a communication map. More details about the malware network connection and propagation could be elicited from network firewall and IDS logs; however, in these experiments we have used only host logs. In this case, the malware infection started from Host one on network A. Host one has access to both networks A and B, and was found to infect hosts two, three and four in network A. The intrusion also propagated from Host one to Host five on network B and then the process continued to all hosts in all networks.

5 Conclusion

This work gave a brief introduction into Microsoft Windows system restore points, and reiterated the value of registry snapshot analysis. A method to infer malware infection times from snapshot data has been given. Infection time information for multiple hosts can then be correlated to derive the malware intrusion attack paths from one host to another, or from one network to another. A case study was then given that shows the practicality of the described method. Overall, the described method gives good results in determining the initial infection time of a host (within 24 hours), which could provide valuable information when attempting to determine current network vulnerabilities or the vector and intent of the initial intrusion. Our in progress work includes improvements in the procedures of traces extraction from Windows restore point to better understand the actions and behavior of malicious code in the infected environment.

References

1. Symantec. (2010). Internet Security Threat Report (Vol. 16).
2. Gladyshev, P. and Patel, A. (2005). *Formalizing Event Time Bounding in Digital Investigations*. International Journal of Digital Evidence, 4(2).
3. Zhu, Y., J. James and P. Gladyshev. (2009). *A comparative methodology for the reconstruction of digital events using Windows Restore Points*. Paper presented at the Digital Investigation Conference.
4. Microsoft. (2010). *About System Restore*, Retrieved 2011, from http://msdn.microsoft.com/en-us/library/aa378724(v=vs.85).aspx
5. K, Harms. (2006). *Forensic analysis of System Restore points in Microsoft Windows XP*. Digital Investigation 3, 151-158.
6. Carvey, H. (2009*). Windows Forensic Analysis DVD ToolKit*.
7. Kahvedzic, D. and Kechadi, T. (2008). *Extraction of User Activity through Comparison of Windows Restore Points.* Citeseer.
8. Kahvedzic, D. and T. Kechadi. (2009). *On the persistence of deleted windows registry data structures*. Paper presented at the ACM symposium on Applied Computing, Honolulu, Hawaii.

9. TechNet, Microsoft. (2002). *Windows XP System Restore*, 2011, from http://technet.microsoft.com/en-us/library/bb490854.aspx
10. Microsoft. (2010). *Monitored File Name Extensions*, Retrieved 2011, from http://msdn.microsoft.com/en-us/library/aa378870(v=vs.85).aspx
11. Microsoft. (2011). *Microsoft PE and COFF Specification*, Retrieved 2011, from http://msdn.microsoft.com/en-us/windows/hardware/gg463119.aspx
12. Zhu, Y. and P. Gladyshev,. (2009). *Temporal Analysis of Windows MRU registry Keys*. Advances in Digital Forensics 306, 83-93.
13. Ammann, P. and Wijesekera, D. and Kaushik, S. (2002). *Scalable, graph-based network vulnerability analysis.* Paper presented at the 9th ACM conference on Computer and communications security, Washington, DC, USA.
14. Ingols, K., Lippmann, R., and Piwowarski, K. (2006). *Practical Attack Graph Generation for Network Defense*. Paper presented at the Annual Computer Security Applications Conference
15. Sheyner, O., Haines, J., Jha, S., and Lippmann, R. (2002). *Automated Generation and Analysis of Attack Graphs*. Paper presented at the IEEE Symposium on Security and Privacy, Los Alamitos, CA, USA.
16. http://www.offensivecomputing.net/
17. http://www.nepenthespharm.com/
18. AccessData (2010). "*Forensic Toolkit*." Retrieved 4 Nov., 2010, from http://www.accessdata.com/forensictoolkit.html.