THE JOURNAL OF DIGITAL FORENSICS, SECURITY AND LAW

Journal of Digital Forensics, Security and Law

Volume 12 | Number 1 Article 8

3-31-2017

Anti-Forensic Trace Detection in Digital Forensic Triage Investigations

Kyoung Jea Park
Woosuk University, Samnye-eup, Wanju-gun, Jeollabuk-do, South Korea

Jung-Min Park
Soon Chun Hyang University, Shinchang-myeon, Asan-si, South Korea

Eun-jin Kim
Pukyong National University, Yongso-ro, Nam-gu, Busan, South Korea

Chang Geun Cheon
Hanyang University, Wangsimni-ro, Seongdong-gu, Seoul, South Korea

Joshua I. James
Legal Informatics and Forensic Science Institute, Hallym University, joshua.i.james@pm.me

Anti-Forensic Trace Detection in Digital Forensic Triage Investigations

Cover Page Footnote

This research is funded by the Korean Information Technology Research Institute (KITRI) under the 'Best of the Best' Information Security Education Initiative.

ANTI-FORENSIC TRACE DETECTION IN DIGITAL FORENSIC TRIAGE INVESTIGATIONS

Kyoung Jea Park[1], Jung-Min Park[2], Eun-jin Kim[3], Chang Geun Cheon[4], Joshua I. James[5]

[1]Woosuk University, Samnye-eup, Wanju-gun, Jeollabuk-do, South Korea
[2]Soon Chun Hyang University, Shinchang-myeon, Asan-si, South Korea
[3]Pukyong National University, Yongso-ro, Nam-gu, Busan, South Korea
[4]Hanyang University, Wangsimni-ro, Seongdong-gu, Seoul, South Korea
[5]Legal Informatics and Forensic Science Institute, Hallym University, Chuncheon, South Korea
joshua.i.james@hallym.ac.kr

ABSTRACT

Anti-forensics, whether intentionally to disrupt investigations or simply an effort to make a computer system run better, is becoming of increasing concern to digital investigators. This work attempts to assess the problem of anti-forensics techniques commonly deployed in South Korea. Based on identified challenges, a method of signature-based anti-forensic trace detection is proposed for triage purposes that will assist investigators in quickly making decisions about the suspect digital devices before conducting a full investigation. Finally, a prototype anti-forensic trace detection system is given to demonstrate the practicality of the proposed method.

Keywords: Anti-Forensics Detection; Digital Forensic Triage; Trace Signature Detection; Preliminary Digital Forensic Analysis; Advanced Preview; Anti-Anti-Forensics; File System Analysis; Windows Registry Analysis

1. INTRODUCTION

With the growing number, type and complexity of digital devices, the amount of data needing analyzed for digital evidence is constantly growing (Casey, Ferraro, & Nguyen, 2009; Gogolin, 2010). With it, there is a growing concern about the use of anti-forensic techniques that attempt to hinder digital investigations (Wundram, Freiling, & Moch, 2013). While there are various motives for anti-forensics (Harris, 2006; Garfinkel, 2007), generally anti-forensic techniques are used to obstruct the acquisition, analysis or validation of digital evidence. For example, there have been a number of cases where disk encryption either prevented further investigation, or proved to be a difficult obstacle to acquiring evidence (Casey, Fellows, Geiger, & Stellatos, 2011; Conrad, Dorn, & Craiger, 2010).

Rogers (Rogers, 2005) defined four categories of anti-forensics; data hiding, artifact wiping, trail obfuscation and attacks against computer forensics. When any of these types of anti-forensic techniques are used, investigators face at least two challenges. The first is simply detecting that some form of anti-forensic technique has been used on a system under investigation. The next is the attempted reconstruction of data or information that was affected by the use of the anti-forensic technique. This work is primarily concerned with the detection of the use of an anti-forensic technique in order to give an investigator more information prior to conducting a full digital forensic investigation. If an investigator can more effectively check whether anti-forensic techniques have been used on a suspect system, such information could poten-

tially influence the strategies used during a full investigation.

Providing more information to an investigator before a full investigation is similar to the objective of triage (Koopmans & James, 2013) or advanced preview (Shaw & Browne, 2013) investigation models discussed in prior works. A number of tools have implemented different anti-forensic detection mechanisms, usually focusing on one category of anti-forensics, which can take too long to be suitable for triage purposes, eg. in the case of file entropy testing to detect encrypted containers. This work instead attempts to apply anti-forensic detection that is fast enough to be suitable for digital forensic triage purposes, where triage provides intelligence for decision making about exhibits, not exhibit exclusion.

This work extends the that of Geiger (Geiger, 2005) who analyzed a number of 'counter-forensic' tools related to artifact wiping. He identified a number of "failure areas" where artifact wiping tools failed to completely remove relevant data. Further, such artifact wiping tools also were found to lead to information disclosure about that tool's usage. Based on these findings, a detection utility was later created that "searches for signatures of tested counter-forensic tools" (Geiger, 2006). This utility uses Regular Expressions to "match patterns in the name fields of deleted MFT records and from associated data sectors". Unfortunately, the signatures that are created and used by the utility are not specifically defined, and the utility itself does not appear to be available, or may be available to Law Enforcement only. Other challenges with this approach include a focus only on artifact wiping tools, where other categories of anti-forensics are neglected; a focus only on NTFS file systems; and an apparent manual creation of signatures using AccessData's Forensic Tool Kit (FTK).

Similar to the work of Geiger, we apply prior automated event reconstruction techniques that utilize signatures based on file system and Windows Registry traces. For example, the work of Khan, Chatwin et al. (Khan, Chatwin, & Young, 2007) attempted to learn application 'footprints' using Bayesian networks based on file system meta-data. A non-probabilistic model was later proposed by James, Gladyshev et al. (James, Gladyshev, & Zhu, 2011) that used real-time system analysis to create signatures of user actions based on created observable traces. Similar methods implementing snapshot-based signature derivation have also been proposed for automatic event reconstruction purposes (Kang, Lee, & Lee, 2013; Kalber, Dewald, & Freiling, 2013). Event reconstruction, however, focuses more on the reconstruction of events in *time*, where this work is concerned only with the detection of traces that may indicate anti-forensic techniques were used on a suspect system.

1.1 Contribution

This work contributes to the field of digital forensic investigation first by assessing the state of anti-forensics techniques encountered in practice by Law Enforcement in South Korea. Based on identified practical needs of investigators, a modified, more robust, real-time signature creation algorithm that allows for the understanding of differences in user actions relating to a specific program is proposed, followed by a novel application of the proposed signature-detection method. Instead of applying signature-based detection to problems of automatic event reconstruction, this work utilizes fast trace detection using derived signatures to assist with digital forensic triage tasks. Finally, this work demonstrates a prototype trace detection system that allows an investigator to quickly triage suspect devices based on traces of anti-forensic activities. The prototype uses a signature structure that can be applied to any file system, and is released as a free, open-source project for use by all.

2. MOTIVATION

While several prior works have examined the problem of anti-forensics from the perspective of classification and detection (Rogers, 2005; Harris, 2006; Garfinkel, 2007; Rekhis & Boudriga, 2010; Wundram et al., 2013), informal discussion with investigators in various countries re-

veal that some forms of anti-forensics, such as advanced data hiding, are not as often encountered in what they consider 'normal' cases. Instead, it appears to be more common for investigators to encounter attempts at artifact wiping.

To identify the state of anti-forensics, and the challenges it poses to practical digital forensic investigations in South Korea, a survey was distributed to Korean public and private sector digital forensic investigators. In total, 11 responses were received[1].

Five investigators claimed to have 3 or more years experience, while six investigators claimed to have less than 3 years experience. 82% (9 out of 11) of respondents claimed to have encountered some form of anti-forensics during their time as a digital forensic investigator. 55% (6 out of 11) of respondents claimed to use some form of anti-forensic detection tool. 100% (11 out of 11) respondents believe there is a need for more-advanced anti-forensic detection tools. Likewise, 100% of investigators listed ease of use as the main desired criteria for an anti-forensic detection tool. Investigators primarily claimed that detection should focus on whether anti-forensic tools exist(ed) on the suspect system, and to what extent they had been used, e.g. installation only, portable, running, uninstalled, etc.

Based on the survey results, there is a need for an easy to use anti-forensic detection tool to help an investigator quickly determine to what extent anti-forensic techniques may have been used on a system under investigation.

3. ANTI-FORENSIC SIGNATURE CREATION AND DETECTION

A *signature* is defined as a list of traces created in a system that are associated with a particular anti-forensic tool or technique. For example, when running an anti-forensic tool in a Windows system, a number of data sources, such as file content or meta-data and Registry entries may be updated. A signature is the collection of these updates, where each update constitutes one 'trace'.

3.1 Signature Creation

Prior works focus on the detection of specific trace update patterns derive signatures either by using snapshot analysis (Kang et al., 2013; Kalber et al., 2013) comparing the updates to data sources between to different snapshots of the same system, or using real-time trace update detection (James et al., 2011).

This work proposes an advancement to the real-time trace update detection. Using real-time trace update detection, it is possible for some specific actions to be differentiated, for example 'install' and 'run' actions, and corresponded to their specific trace update patterns. Similar to prior works, trace updates are detected from the file system and Windows Registry, if available.

In this work, signatures of anti-forensics were created using the following method:

1. Create test system (Virtual Machine)
2. Run file system logger (Process Monitor)
3. Execute desired action
   - Install
   - Run/Execute Anti-Forensic Technique
   - Uninstall
4. Save file system logger output
5. Filter log to reduce noise
6. Extract usable unique signature
7. Define traces in resulting signature as regular expressions for portability

This method differs from (James et al., 2011) in two ways. First, specific actions – such as install, run and uninstall – are targets for signature creation rather than grouping all actions into one overall signature. This means that if unique signatures for each type of action can be

[1]Survey questions and results can be found at http://www.cybercrimetech.com/2013/12/bob-indicators-of-anti-forensics.html

derived then identification in the use of an anti-forensic tool may be possible. Next, filtering is more aggressive. Because reconstruction of events in time is not necessary, only the traces that are unique to the action are necessary.

3.1.1 Create Test System

In this work, signature creation was implemented using a virtual machine running a 64bit version of Windows 7. Process Monitor (procmon) was installed on the system for real-time file system and Windows Registry update monitoring. A snapshot of the 'clean' system was then created for easy system rollback after testing. Such an environment could potentially be used to create signatures relating to any operating system.

At this stage, the monitoring program was used to create a baseline system activity log. Monitoring was enabled on the system for 5 minutes with no user activities running. The result is a log of system activities that can be considered as noise. This log was labeled as 'falselog.xml' and will be used to filter results from application-specific logs.

3.1.2 Monitor, execute and save

Once a test system has been created, the action to test must be determined. In this case, the focus is on anti-forensic programs. Signatures were created for the actions install, run/execute, and uninstall. If the program was 'portable' or does not need to be installed, then install and uninstall was skipped (Figure 1).

For each selected program, the file system (and Registry) monitor was started, and each action relating to the specific program was executed. After each action was executed, the monitor was stopped, the log was exported, and the log buffer was cleared. Monitoring would be started again, and the next action in the sequence would be executed.

After all actions in the sequence were executed, and logs were collected, the virtual machine was revered back to the original snapshot. This process was completed five times per identified application. The resulting logs were a collection of XML files with the name of the analyzed anti-forensics tool, and the action that was recorded.

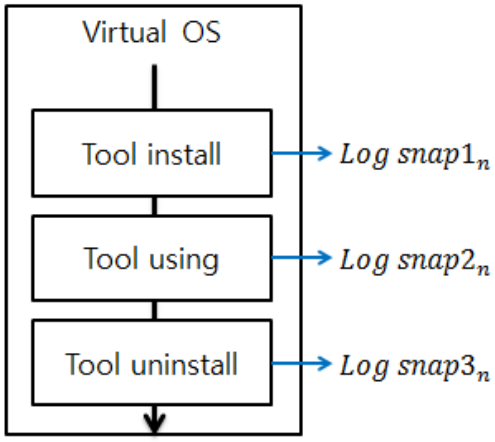

Figure 1. The action log collection process executed five times per anti-forensics application

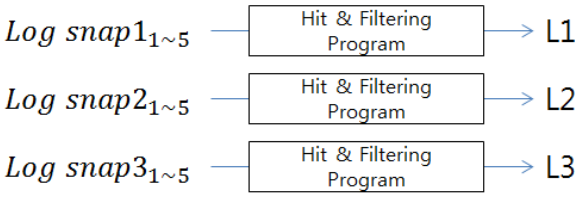

Figure 2. Action logs for specific anti-forensic programs combined to filter non-consistently updated traces per action type

3.1.3 Filter log to reduce noise

The result of the prior step is five logs per action per anti-forensic program. Since this work is concerned with reliable traces of anti-forensic activities, filtering is used to extract only commonly-created traces. A filtering program was created to count the number times a particular trace was updated for a given action (Figure 2. An excerpt of the results are shown in table 1. Traces that were not updated at least once per action log were discarded. Further, any traces that exist in both the action log, and the previously-created 'falselog.xml' are considered as background noise, and were removed from the action log.

Once system and other background 'noise' have been removed, the resulting list of traces are compared to the 'clean' test system. Any

Process	Operation	Path	Total Hit	Log0 Hit	file1 Hit	Log2 Hit	Log3 Hit	Log4 Hit
Explorer.EXE	ReadFile	C:\Program Files\Eraser\Eraser.exe	25	5	5	5	5	5
Explorer.EXE	RegOpenKey	HKCU\Software\Classes\Applications\Eraser.exe	22	5	4	4	4	6
Explorer.EXE	RegOpenKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Eraser.exe	5	1	1	1	1	1
Explorer.EXE	RegOpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\Eraser.exe	5	1	1	1	1	1
Explorer.EXE	RegOpenKey	HKCR*\shellex\ContextMenuHandlers\Eraser	1	1	1	1	1	1

Table 1. A selection of filtered traces relating to the action 'run' for the anti-forensic program 'Eraser'

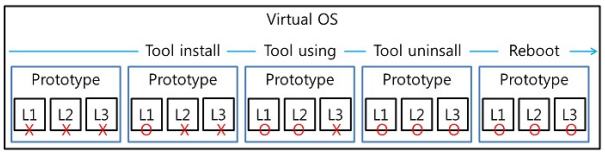

Figure 3. Traces of actions detected after each action was executed, where L1 refers to installation, L2 refers to execution, L3 refers to uninstallation, where circle means a trace of the action was detected and X means no trace of the action was detected

Action	Trace
Install	C:\Users\user\AppData \LocalLow\Microsoft\CryptnetUrlCache \MetaData \CE4CFAB51DB3F9AB265C1526D1E6F12F_ FC8C5CB969BCDC8ACE4FEF989663C7A4
Run	HKCU\Software\Eraser\Eraser 6\9977d7c4-c940-4b73-a02a-33c9ee2d47fe
Uninstall	C:\Users\user\AppData\LocalLow\Microsoft \CryptnetUrlCache\Content \CE4CFAB51DB3F9AB265C1526D1E6F12F_ FC8C5CB969BCDC8ACE4FEF989663C7A4

Table 2. A selection of unique traces for each identified action relating to the anti-forensic program 'Eraser'

traces that are detected on the clean test system are considered false positives, and are removed. The resulting list are traces that are specific to the action conducted with the application. However, they may not be unique to the action. Each trace may be updated by either another action relating to the same application, or may potentially overlap with other currently-unknown applications. In prior works of event reconstruction, unknown applications are a problem. However, with digital forensic triage this is a non-issue since any overlap would, at worst, cause a false positive that an investigator would need to manually verify during their full investigation.

3.1.4 Extract usable unique signature

Once the list of traces have been filtered for noise and false positives, traces specific to the action should be determined. This is done in two ways. First, for each action relating to a specific anti-forensic tool, the resulting logs can be compared. Any traces that exist in both logs may be removed.

Another option is to execute many actions in the test system that are not related to the action in question. After the execution of other non-related actions, if any traces in the signature match, these should also be considered false positives.

The result of this process should be a short list of traces that are specific to the action to be tested. Once this short list is created for all actions relating to the particular anti-forensics tool, then the process of installation, execution and uninstallation should be executed on the test system, and each resulting signature should be tested after each action. If traces in a signature match before their associated action has occurred, these traces should be removed.

One example of signature testing is shown in figure 3. Here it is shown that before any actions associated with the anti-forensic tool are executed, none of the resulting signatures match. After installation, only the installation signature (L1) matches; after the anti-forensic tool is executed both the installation (L1) and execution (L2) signatures match. After uninstall, all three action signatures match, and this match is persistent after reboot.

Using this method, the traces specific all actions that are persistent even after a reboot can be identified. A selection of unique traces for each action associated with the anti-forensic program 'Eraser' is given in table 2.

3.1.5 Define traces in resulting signature as regular expressions for portability

As discussed in prior work (James et al., 2011; Kang et al., 2013) some form of generalization of traces within signatures needs to take place to

allow for detection on other systems. This work uses regular expressions to generalize variables in signatures. Implementation, as shown, uses the path and file name, or Registry key. Regular expressions are used for fields that are likely to change depending on system settings, while keeping the path name as specific as possible to ensure only the identified trace is returned by the regular expression. This will enable the same signatures to be used on similar systems, however, it should be noted that signatures are likely to be different depending on the operating system, and perhaps even the version of the anti-forensic program.

4. SIGNATURE MATCHING

To illustrate the practicality of trace detection using pre-defined action signatures for digital forensic triage purposes, a prototype signature matching tool was created. The program detects what are defined as 'Indicators of Anti-Forensics' (IOAF) to help a digital investigator make decisions about suspect systems in regards to anti-forensic activity. The IOAF tool is available under a free, open source license[2].

The IOAF detection prototype currently accepts a forensic disk image as an input, along with a set of signatures (Regular Expressions) to be detected (Figure 4). The MFT parsing module uses 'fls' in the Sleuth Kit to parse the input disk image, and outputs the file system information to a SQLite database. The Sleuthkit (icat) is also used to extract Registry hives from an input disk image. Keys and their associated values are also saved to a separate SQLite database. Signatures are stored as regular expressions in a separate signature database.

After parsing the file system and Registry (if available) with the Sleuth Kit, the signature parsing and matching module compares traces in the signature database with traces stored in the file system and Registry databases. The success or failure of a defined trace matching one (or more - in the case of deleted files) is output to a report file. A successful match (traces of anti-forensic activities) is shown in green, while a non-matching pattern is shown in red (no indication of anti-forensics). Example output reports for the anti-forensic tool 'Eraser', and the actions 'install' and 'run' are shown in figures 5 6 7. Figure 5 shows derived signatures tested against a system where the actions relating to the anti-forensic tool have not been run. In this case, no indicators of anti-forensics are found. Figure 6 shows the same signature tested after the action 'install' has been run on the suspect system. In this case only the install action has associated indicators that are detected. Finally, figure 7 shows the same signature tested after running both the install and execute actions, where both actions resulted in detectable indicators.

[2]The source for IOAF is available at https://github.com/CheonChangGeun/IOAF

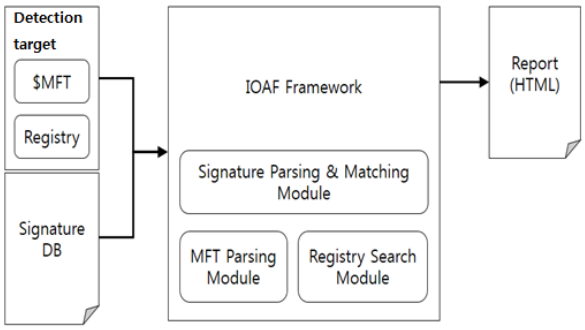

Figure 4. Design of the Indicators of Anti-Forensics prototype signature-matching program

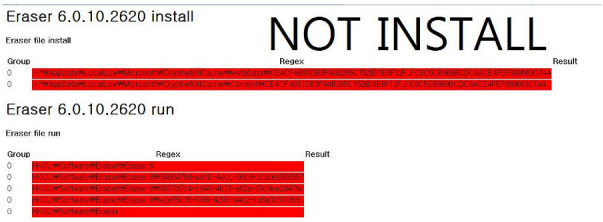

Figure 5. The Indicators of Anti-Forensics prototype resulting report when no traces defined in the signature database are found in the suspect system

4.1 Weaknesses

There are a number of weaknesses with the proposed method. One that is common to all

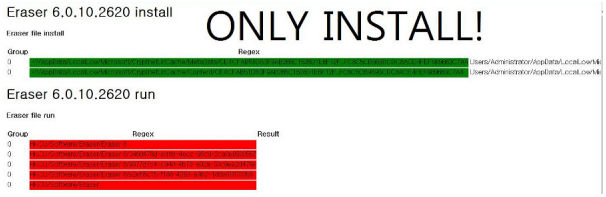

Figure 6. The Indicators of Anti-Forensics prototype resulting report when one action defined in the signature database is found in the suspect system (green), and indicators of other actions are not found (red)

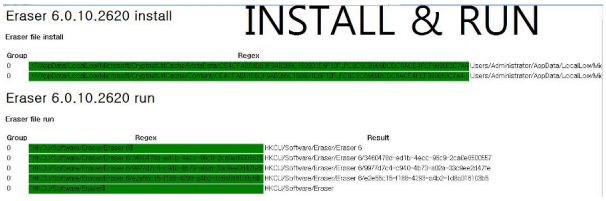

Figure 7. The Indicators of Anti-Forensics prototype resulting report when all traces defined in the signature database are found in the suspect system

signature-based detection methods is that if a signature does not exist for each specific anti-forensic technique, then the technique cannot be detected. Some prior knowledge about the anti-forensic technique is required to be able to derive a related signature. Once a signature is derived, however, it is can be easily shared between investigators. More global collaboration between investigators – to create and share signatures of found anti-forensic techniques – could potentially reduce this weakness.

Similarly, as anti-forensic techniques or specific tools are developed over time, the resulting associated traces will also change. While there normally appears to be a core group of relatively generic traces (James et al., 2011), signatures for anti-forensic techniques and programs will need to be maintained over time.

The proposed method, in an attempt to be a fast triage tool, currently only utilizes file metadata and the current Windows Registry. More data sources, such as Windows Restore Points and log files, should be included in the analysis. This is partially a weakness with the proposed method for creating signatures. The current method examines what files are affected by anti-forensic actions, but is not suitable for determining exactly what changes were made to the contents of the files, except in the case of the Windows Registry.

5. CONCLUSIONS

Digital investigators, at least within South Korea, are encountering the use of anti-forensic tools and techniques. Although it is difficult to determine the extent of the problem, investigators do see a need for better detection when such techniques are used on systems under investigation. This work has proposed a method for generally identifying whether anti-forensic tools exist, and – in some cases – to what extent those tools have been used. By focusing on anti-forensic trace detection rather than full event reconstruction, such a method can be implemented as a type of digital investigation triage tool that quickly gives an investigator more information about suspect systems that have yet to receive a full analysis. This can help investigators prioritize devices, as well as ensuring investigators are better informed about the potential state of a suspect device.

This work has demonstrated the 'Indicators of Anti-Forensics' detection tool. The results of the tool are dependent on the quality of the signatures that are created, whether unique signatures for each 'action' exist, and whether a signature for an anti-forensic tool or technique can be created at all. Some techniques, for example, may leave no discernible traces on a system. While the absence of information could be an indicator of anti-forensic activities, if changes to the system are consistent with the normal running of the system, then a signature of the action can be difficult or impossible to create. Similarly, overly general trace definitions may result in multiple traces being detected that are false positives. While false positives are preferred over false negatives in digital investigations, ensuring the quality of signatures can help to reduce these challenges.

Future work will first look at improving the

user experience of the IOAF detection tool for faster trace detection and better processing and reporting within a digital forensic triage workflow. Aside from technical aspects, future work will include a better understanding of the needs of Law Enforcement outside of South Korea in terms of anti-forensics detection.

ACKNOWLEDGMENTS

This research is funded by the Korean Information Technology Research Institute (KITRI) under the 'Best of the Best' Information Security Education Initiative.

REFERENCES

Casey, E., Fellows, G., Geiger, M., & Stellatos, G. (2011, November). The growing impact of full disk encryption on digital forensics. *Digital Investigation*, *8*(2), 129–134. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1742287611000727 doi: 10.1016/j.diin.2011.09.005

Casey, E., Ferraro, M., & Nguyen, L. (2009). Investigation Delayed Is Justice Denied: Proposals for Expediting Forensic Examinations of Digital Evidence. *Journal of forensic sciences*, *54*(6), 1353–1364. Retrieved from http://www3.interscience.wiley.com/journal/122599763/abstract doi: 10.1111/j.1556-4029.2009.01150.x

Conrad, S., Dorn, G., & Craiger, P. (2010). Forensic Analysis of a PlayStation 3 Console. In *Advances in digital forensics vi* (pp. 65–76). Springer Berlin Heidelberg. doi: 10.1007/978-3-642-15506-2_5

Garfinkel, S. (2007). Anti-forensics: Techniques, detection and countermeasures. In *The 2nd international conference on i-warfare and security (iciw)* (pp. 77–84).

Geiger, M. (2005). Evaluating Commercial Counter-Forensic Tools. *DFRWS*, 1–12. Retrieved from https://www.dfrws.org/2005/proceedings/geiger_couterforensics.pdf

Geiger, M. (2006). Counter-forensic tools: Analysis and data recovery. *18th FIRST Conference*. Retrieved from http://www.cms.first.org/conference/2006/papers/geiger-matthew-papers.pdf

Gogolin, G. (2010, October). The Digital Crime Tsunami. *Digital Investigation*, *7*(1-2), 3–8. Retrieved from http://www.sciencedirect.com/science/article/B7CW4-50S2DC9-1/2/1c6a6a38b9f633ddcd445b2115739ac7http://linkinghub.elsevier.com/retrieve/pii/S1742287610000526 doi: 10.1016/j.diin.2010.07.001

Harris, R. (2006, September). Arriving at an anti-forensics consensus: Examining how to define and control the anti-forensics problem. *Digital Investigation*, *3*, 44–49. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1742287606000673 doi: 10.1016/j.diin.2006.06.005

James, J. I. J., Gladyshev, P., & Zhu, Y. (2011). Signature Based Detection of User Events for Post-Mortem Forensic Analysis. *Digital Forensics and Cyber Crime*, *53*, 96–109. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-19513-6_8http://arxiv.org/abs/1302.2395 doi: 10.1007/978-3-642-19513-6_8

Kalber, S., Dewald, A., & Freiling, F. C. (2013, March). Forensic Application-Fingerprinting Based on File System Metadata. In *2013 seventh international conference on it security incident management and it forensics* (pp. 98–112). IEEE. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6568558 doi: 10.1109/IMF.2013.20

Kang, J., Lee, S., & Lee, H. (2013). A Digital Forensic Framework for Automated User Activity Reconstruction. In R. H. Deng & T. Feng (Eds.), *Information security*

practice and experience (pp. 263–277). Springer Berlin Heidelberg. doi: 10.1007/978-3-642-38033-4_19

Khan, M. N. A., Chatwin, C. R., & Young, R. C. D. (2007). Extracting Evidence from Filesystem Activity using Bayesian Networks. *International journal of Forensic computer science*, *1*, 50–63. Retrieved from http://www.ijofcs.org/V02N1-P04-ExtractingEvidencefromFilesystem.pdf

Koopmans, M. B., & James, J. I. (2013, September). Automated network triage. *Digital Investigation*, *10*(2), 129–137. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1742287613000273 doi: 10.1016/j.diin.2013.03.002

Rekhis, S., & Boudriga, N. (2010). Formal Digital Investigation of Anti-forensic Attacks. In *Fifth international workshop on systematic approaches to digital forensic engineering* (pp. 33–44). IEEE Computer Society. doi: http://dx.doi.org/10.1109/SADFE.2010.9

Rogers, M. K. (2005). Ant-Forensics. In *Lockheed martin.* San Diego, California. Retrieved from http://cyberforensics.purdue.edu/documents/AntiForensics_LockheedMartin09152005.pdf

Shaw, A., & Browne, A. (2013, September). A practical and robust approach to coping with large volumes of data submitted for digital forensic examination. *Digital Investigation*, *10*(2), 116–128. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1742287613000327 doi: 10.1016/j.diin.2013.04.003

Wundram, M., Freiling, F. C., & Moch, C. (2013, March). Anti-forensics: The Next Step in Digital Forensics Tool Testing. In *2013 seventh international conference on it security incident management and it forensics* (pp. 83–97). IEEE. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6568557 doi: 10.1109/IMF.2013.17

