Identifying Malware Using Cross-Evidence Correlation

Anders Flaglien, Katrin Franke, Andre Arnes

Chapter 13

IDENTIFYING MALWARE USING CROSS-EVIDENCE CORRELATION

Anders Flaglien, Katrin Franke and Andre Arnes

Abstract This paper proposes a new correlation method for the automatic identification of malware traces across multiple computers. The method supports forensic investigations by efficiently identifying patterns in large, complex datasets using link mining techniques. Digital forensic processes are followed to ensure evidence integrity and chain of custody.

Keywords: Botnets, malware detection, link mining, evidence correlation

1. Introduction

Rapidly growing data volumes, increasing computer system complexity and obfuscated malware make forensic investigations of malware cases time consuming, costly and ever more difficult [14]. Botnets, in particular, pose serious threats [19, 22]; they utilize malware to establish control over infected machines. However, due to the command and control architecture of botnets, evidence is present in multiple locations. This requires the use of correlation techniques for forensic investigations of botnet infections.

The architecture of forensic tools limits their utility in analyzing large, complex datasets from multiple computer systems. Investigating cases involving such datasets (e.g., in botnet incidents) often requires substantial and time-consuming manual analysis [3, 5, 10, 24].

This paper proposes a digital forensic correlation method for malware-related evidence that automates the analysis of large, complex datasets from multiple computer systems. Three key issues are investigated: (i) the features that can be used to correlate and identify malware traces; (ii) the application of correlation techniques; and (iii) the impact of

the correlation techniques on the effectiveness and efficiency of digital forensic investigations involving malware.

2. Related Work

The research of Garfinkel and others [3, 7, 10, 11] on automating digital forensic analysis is a good starting point for our discussion of related work. Of particular interest is Garfinkel's cross-drive analysis methodology [10] for detecting extracted features (e.g., email addresses, social security numbers and credit card numbers) present in multiple hard drives. The methodology demonstrates the benefits of correlating evidence, but suffers due to its use of a limited set of features.

It is difficult, if not impossible, to manually identify digital evidence in large data volumes. Substantial research has focused on improving the effectiveness and efficiency of this task. However, state-of-the-art digital forensic suites such as EnCase and FTK are generally unable to efficiently process large volumes of data [3].

Case, *et al.* [5] have developed FACE, a framework for discovering and correlating evidence from multiple components in a single computer. To combine multiple evidence sources, all the data must be represented in the same format. Garfinkel [11] has designed a common representation format for evidence from multiple sources. The Fiwalk program can be used to analyze data structures and extract file attributes represented in XML and ARFF (Attribute Relationship File Format). ARFF is especially interesting due to its support in data mining tools such as Weka [16]. Interested readers are referred to [9] for a comparative analysis of digital forensic storage and exchange formats.

Mena [21] describes data mining techniques for investigating security breaches and other incidents by correlating evidence. Link analysis, in particular, is a powerful approach for modeling links between entities associated with physical crimes as well as security incidents.

Malware can be hard to detect because it often uses obfuscation techniques. However, malware such as botnets that require a command and control architecture manifest certain patterns (e.g., bot master control actions). Zeng, *et al.* [29] have achieved good results by combining network and host information to detect botnet activity. Their method correlates evidence effectively, but is limited to live systems.

Al-Hammadi and Aickelin [1] have examined correlations between normal user activity and bot activity on multiple hosts. They used log file information generated after injecting DLLs into processes of interest in multiple machines, some of which were infected with malware. By comparing normal IRC user activity against IRC bot activity, Al-Hammadi

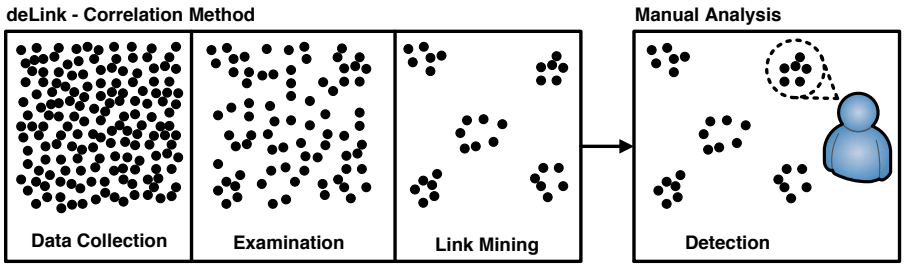

Figure 1. Conceptual view of the deLink method.

and Aickelin discovered that IRC bot activity exhibited much higher correlations than normal user behavior.

Clustering is a powerful tool for identifying common patterns and correlating data. In the context of data mining, clustering has been used in association with group detection [13], for link based cluster analysis [12] and to identify common characteristics of criminal suspects [6]. K-means is one of the most popular clustering algorithms [26, 28]. X-means, an algorithm based on K-means has been used to successfully group bots with common communication and activity patterns [15].

3. Correlation Method

This section describes the deLink method, which is designed to identify malware-related evidence across multiple computers.

Figure 1 presents a conceptual view of the deLink method. The main components are data collection, examination and link mining. The deLink method would typically be applied in a digital forensic investigation where multiple evidence sources are involved. The output of the method is a filtered, structured dataset, which is clustered based on common linked patterns from all involved sources. This enables the characteristics of the linked file objects to be analyzed more efficiently by a forensic investigator, for example, to reveal interesting groups of correlated data as shown in the circle in Figure 1.

3.1 Data Collection

Data collection for deLink involves making a forensically-sound copy of the original media. The main concerns are preserving the integrity of the evidence, i.e., ensuring that the data is preserved in its original form and that it has not been accidentally or willfully manipulated; and maintaining the chain of custody, i.e., documenting the possession and location of evidence at all times. Note that the media type considered in the proof-of-concept deLink implementation is a computer hard drive.

Table 1. Features of interest.

Features	Values
File Metadata	Time stamp, file name, type, allocation status, file system entry, permissions, links, UID, GID, sequence number
Case Metadata	File ID, machine ID, media ID
File Content Based Data	File content type, MD5 hash value, file entropy value, IP addresses, email addresses, URLs

3.2 Data Examination

Because of the potentially large quantities of data that are collected, it is necessary to apply filtering techniques (e.g., based on file hash values) to limit the amount of data to be examined. We employed the well-known NSRL RDS dataset [23] in our work. In addition, a hash dataset based on clean systems with similar configurations as the investigated machines was used. To further improve the quality of the examined data, certain features of interest were extracted from the files that remained after filtering. The features of interest correspond to the important characteristics that can be used to identify malware files. Typical features based on known malware include incident timestamps, file- and system-specific anomalies, keywords and identifiers, and anomaly-obfuscated items.

We have defined three categories of features, file metadata, case metadata and file content based data, based on typical file features [11] and on malware communication characteristics associated with botnets [25], respectively (Table 1). While most of the values are typical file metadata features, the content based features (IP, email and URL addresses) along with the MD5 hash values and the entropy values of file content are based on malware and their communication characteristics. The entropy values are especially relevant for detecting obfuscation (e.g., executable files with higher entropy values than text files). Note that we do not focus on selecting the optimal feature set; optimal feature selection for cross-evidence based malware identification is a topic for future work.

Case metadata features can help distinguish evidence from different computer systems. This improves traceback functionality to the source when many computers yield a large dataset. However, the features are not involved in link mining as they would negatively dominate other features and affect the results. This is because files from the same hard drive always have the same machine ID and media ID.

The ARFF data format is used to represent the features [27]. This file format, which was developed for the Weka machine learning tool [16], represents datasets as independent objects that share a defined set of attributes. A feature extraction tool based on Fiwalk [11] was used to extract and represent file metadata in ARFF. SleuthKit [4] was used to extract content based features required by deLink (e.g., IP, email and URL addresses). In addition, case metadata features were manually added to ARFF.

3.3 Link Mining

Link mining is an emerging discipline of data mining whose goal is to produce a structured presentation of interconnected and linked objects. When links are visualized, one gains a better understanding of the relationships and associations of objects in a particular dataset. Link mining is frequently used to analyze social networks and the World Wide Web (interconnected by hyperlinks); it is also employed in medical research, and financial and bibliographic analysis [13, 21].

deLink uses link mining on the dataset of features to reveal correlations. The features are preprocessed before applying the chosen link mining algorithm. This is done to best suit the clustering task and to remove dominant features. Unsupervised clustering, which is a descriptive data mining method, is used to group files with similar characteristics. This can unite different groups of data with common patterns and identify links existing between them [12, 13]. An unsupervised method is used instead of a supervised method because of the lack of details about specific malware characteristics or signatures that could be used to classify files (e.g., as malicious or insignificant).

K-means is one of the most popular clustering algorithms [26, 28]. The number of clusters (K) must be known in order to use the K-means algorithm. K is based on the natural groups existing in the data, which can be determined using self-organized maps (SOMs). SOM generation is an unsupervised learning technique that creates a two-dimensional grid of cells from multidimensional datasets [21]. Based on a value of K provided by a SOM, K points are randomly selected by the K-means algorithm. The algorithm assigns objects in the dataset to the cluster with the closest center (cluster centroid). The Euclidean distance is used to measure proximity [17]:

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + ... + (x_{in} - x_{jn})^2}$$

Next, the mean values of the objects assigned to the various clusters are calculated iteratively until all the cluster centers have stabilized. In the case of a large, multidimensional dataset, file objects with the most common characteristics are placed in the same cluster. Due to the use of Euclidean distance, there are some limitations regarding dominant features. These must be considered during feature preprocessing and result evaluation [17].

Weka was used to preprocess the ARFF files and to execute the K-means algorithm on the dataset. The visualization features of Weka were used to supplement the analysis of the results.

3.4 Evaluation

deLink examines and preprocesses input data in several stages. The input data includes machine IDs, media IDs, file object IDs and original file locations. In cases where the final clustering results are used to identify files of interest, links to the original file content are also required.

The evaluation of the results obtained through link mining requires special attention. This is because of the possibility of misinterpreting the link mining results [20, 26].

The clustering results used to link objects across machines can be evaluated against a predefined class attribute in the dataset. This evaluation depends on a thorough understanding of the dataset and the ability to correctly classify the data. By comparing the classified data with the clustering results, it is possible to reveal uncertainties regarding the integrity of the clustering task, algorithm and the features used [26].

The links may also be evaluated using two-dimensional graphs of the features involved in the created clusters. These indicate variations within and between the K clusters (C), possibly using the within-cluster variations (wc) and between-cluster variations (bc), which are given by [18]:

$$wc(C) = \sum_{k=1}^{K} wc(C_K) = \sum_{k=1}^{K} \sum_{x_i \in C_K} d(x_i, r_k)^2$$

$$bc(C) = \sum_{1 \leq j < k \leq K} d(r_j, r_k)^2$$

Note that wc measures cluster compactness based on the Euclidean distances between a data point x_i and the cluster centroids r_k. On the other hand, bc measures the distances between the cluster centroids. The presence of a file across the created clusters reflects the correlations among the machines [27].

4. Experiments and Results

The experiments that were conducted focused on botnets involved in online banking fraud [25]. This enhances the realism of the experiments and also brings to bear expert knowledge (e.g., temporal and spatial information) that an investigator would typically use in a case.

Three experiments were conducted to evaluate the performance of the deLink method: (i) Proof-of-concept (one machine); (ii) Keylogger bot malware (multiple machines); and (iii) Spybot v1.3 – "malware from the wild" (multiple machines).

The first two experiments were control experiments that were designed to verify the ability of deLink to successfully extract and represent features, and to identify planted and correlated files across a dataset. Interested readers are referred to [8] for additional details about these experiments. The third experiment involved the use of "malware from the wild" (Spybot v1.3) to infect a group of machines (M_1 – M_5). The first two experiments were executed successfully with the expected results. Consequently, this paper primarily focuses on the third and most significant experiment involving malware from the wild.

4.1 Experimental Setup

All three experiments were executed in virtual environments. Despite certain challenges and limitations, virtualization techniques have yielded positive results in a numerous digital forensic experiments, as in the case of the Virtual Security Testbed ViSe [2]. The machines used in our experiments were configured virtually using VMWare Workstation 7.0.0 build-203739 with a 4 GB hard disk, 512 MB memory and running the Windows XP SP2 operating system.

Instead of installing the machines separately, the clone function in VMware was used to create a clean state for each machine. The malware file was then added to each cloned machine.

Figure 2 clarifies the process: an initial system was created, a clean state was defined and subsequently duplicated (cloned) in all five machines. Each machine was then infected separately. The machines were cloned to ensure that the hash database corresponding to the clean state would be the same for all the machines. Also, using cloned machines makes it possible to remove more data objects than from individually-configured machines. This is not possible in a real-world scenario, but, in this experiment, it decreases the time taken to extract content based features at a later stage and significantly increases the number of resulting file objects associated with the file system of each machine.

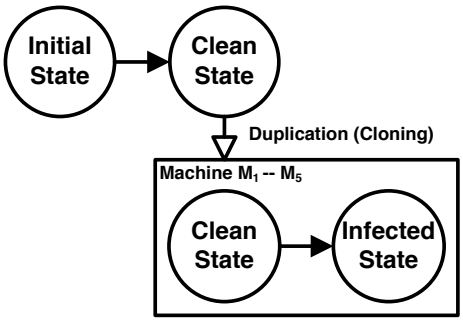

Figure 2. Virtual machine states.

4.2 Processing Steps

Several processing steps are performed by deLink in order to correctly collect, examine, combine, preprocess and identify links. A feature extractor and file parser were developed to automate most of the processing.

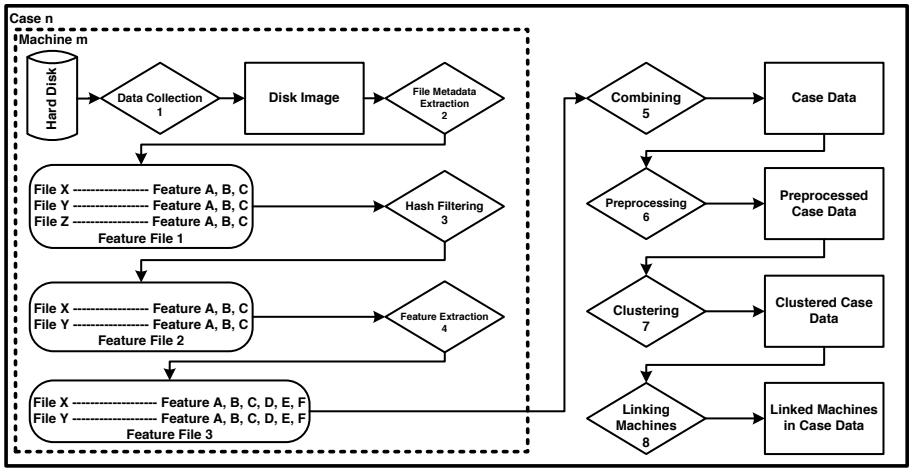

Figure 3. Processing steps for a case.

Figure 3 presents a case involving m machines that were seized as a result of their involvement in an incident. The data from each machine is collected and preserved by creating a Disk Image using the `dcfldd` tool (Step 1). Next, file metadata is extracted from the Disk Image to create Feature File 1 (Step 2). Step 3 involves hash filtering, where known files are removed based on their hash values to create Feature File

Table 2. Number of files before and after filtering.

Machine ID	Initial	Post-Filtering
1	13,871	434
2	13,871	432
3	13,871	431
4	13,871	431
5	13,871	433
Clean	13,867	–
All	69,355	2,161

2. In Step 4, additional content based features are extracted from the filtered metadata representation. User-supplied metadata is also added to separate machines and media and produce a metadata representation with additional file and machine features (Feature File 3).

Step 5 is a manual process that combines the Feature File 3 corresponding to each of the m machines to produce the Case Data of all the machines involved in the case. Step 6 involves preprocessing to obtain the correct representation of each feature for the clustering step (Step 7). This yields the Clustered Case Data, which reveals the machines and files that are correlated. The final step (Step 8) involves the manual linking of machines in the Clustered Case Data by the investigator.

4.3 Examination Results

Automated hash filtering reduces the number of file objects by approximately 97%. The NSRL RDS dataset alone removed 8,916 file objects; the remaining files were filtered using the clean system hashes. Table 2 lists the file objects that remained after each filtering step.

4.4 Link Mining Results

The dataset of feature files extracted from the machines was used to produce an output SOM with three groups (Figure 4). Note that the key parameter is the number of groups, not necessarily the contents of the groups.

Weka was used to partition the dataset into three clusters whose properties are presented in Table 3. Note that Cluster denotes the ID of each clustered data group, Objects denotes the number of files in each cluster and Percentage denotes the percentage of files in each cluster. Little variation was seen in the number of file objects associated with each cluster. This was also true for their characteristics, where clear differences were absent until further analysis was performed.

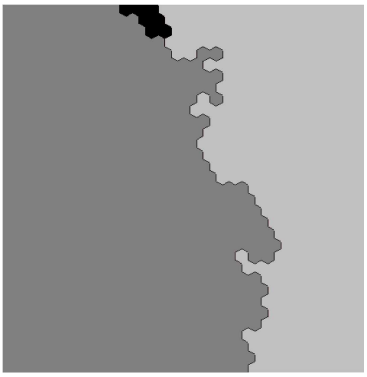

Figure 4. SOM for malware from the wild.

Table 3. Clustered instances.

Cluster	Objects	Percentage
1	604	28%
2	882	41%
3	675	31%
All	2,161	100%

File objects from all five machines are present in the three clusters. Thus, there are links between all the machines, some of them more relevant than others. Expert knowledge about the incident and file system analysis is obviously still necessary to clarify the characteristics of the clusters.

Table 4. Clusters based on temporal and spatial incident information.

Cluster	2010-04-15 20:30-20:35	192.168.40.129
1	Before and after	30 file objects
2	Before and after	None
3	Before and after	None

Temporal and spatial information about the incident were used for further analysis of each cluster. Apart from the botnet infection itself, there was no evidence of the execution of any botnet attacks in the experiment. Thus, no IP address information of victim sites existed, and, therefore, the timestamps for botnet command and control communications and the IP address of the command and control server were used instead. Table 4 provides data pertaining to the clusters.

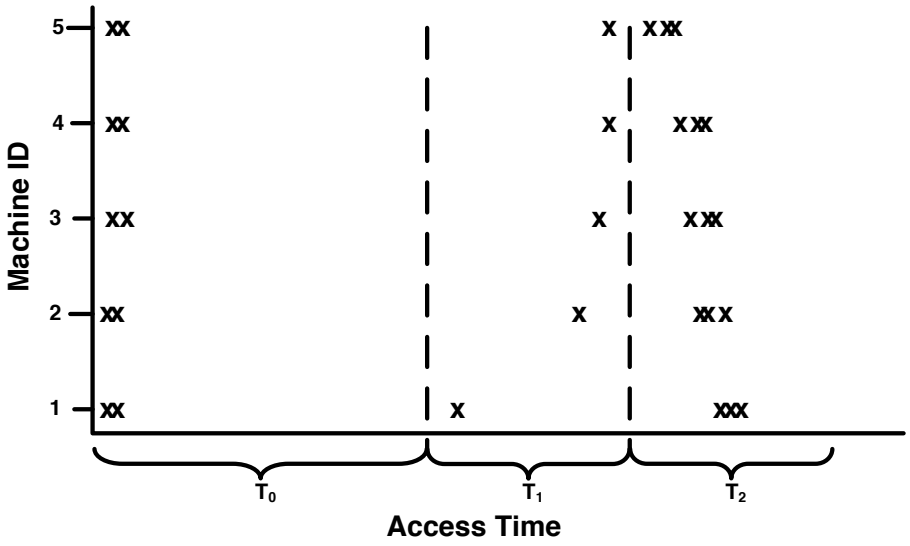

Figure 5. Last access timestamp of Cluster 1 with IP address `192.168.40.129`.

Due to the high frequency of the IP address for command and control communications in Cluster 1, the corresponding file objects were filtered and presented graphically for each of the five machines based on the access time. After performing the filtering, the last access timestamps of the files appeared more clearly. The correlation between the machines, which is shown graphically in Figure 5, further improved the identification of files that were involved in the incident (marked as **x**). The time intervals T_0 – T_2 would typically be defined by an investigator to separate the events.

At time T_0, two Internet Explorer cache file version 5.2 files were accessed in all five machines. Only one file is visible in Figure 5 because of the identical access times. The two files were `index.dat`, which were located at `...\History\History.IE5\` and `...\Temporary Internet Files\Content.IE5\`.

At time T_1, another Internet Explorer cache file version 5.2 file from the `...\History\History.IE5\` folder was accessed in all five machines.

At time T_2, multiple files with suspicious characteristics were accessed from all five machines sequentially, within approximately one minute (`20:32:48` to `20:34:01`).

The files were suspicious because of their content (PE32 executable for MS Windows (GUI) Intel 80386 32-bit) and their relatively high entropy value of 6,218,053 compared with the other files in the cluster. One of the files was `spyware.exe`, which was located in the `...\Temporary`

`Internet Files\Content.IE5\` folders of the machines. The other two files were located at `\system32\`, a well-known location for hiding malware. These files, `wuaumqr.exe` and `download_me.exe`, seemed suspicious; a Google search verified that they are often associated with malware. The latter file was stored under `\system32\kazaabackupfiles\` that was also verified as a location for hiding malware.

5. Conclusions

The correlation method described in this paper supports forensic investigations by efficiently identifying patterns in large, complex datasets using link mining techniques. Despite the simplified experiments, the results demonstrate that the method facilitates the detection of correlations in evidence existing on the hard drives of multiple machines. In addition, the content based file features, especially IP addresses, along with the entropy values, timestamps and the type of file content, help identify malware-related evidence.

Additional experiments should be conducted to evaluate the correlation method; these experiments should be conducted on a variety of machines, storage device types and file systems. In addition, a heterogeneous distribution of infected and uninfected machines from multiple locations and environments should be tested. Finally, theoretical and experimental analyses of approaches for feature selection, distance measures and clustering should be undertaken to enable investigators to choose the best techniques for correlating evidence in large, complex datasets from multiple computers.

References

[1] Y. Al-Hammadi and U. Aickelin, Detecting botnets through log correlation, *Proceedings of the Workshop on Monitoring, Attack Detection and Mitigation*, 2006.

[2] A. Arnes, P. Haas, G. Vigna and R. Kemmerer, Using a virtual security testbed for digital forensic reconstruction, *Computer Virology*, vol. 2(4), pp. 275–289, 2007.

[3] D. Ayers, A second generation computer forensic analysis system, *Digital Investigation*, vol. 6(S), pp. 34–42, 2009.

[4] B. Carrier, The Sleuth Kit (www.sleuthkit.org).

[5] A. Case, A. Cristina, L. Marziale, G. Richard and V. Roussev, FACE: Automated digital evidence discovery and correlation, *Digital Investigation*, vol. 5(S), pp. 65–75, 2008.

[6] H. Chen, W. Chung, J. Xu, G. Wang and Y. Qin, Crime data mining: A general framework and some examples, *IEEE Computer*, vol. 37(4), pp. 50–56, 2004.

[7] M. Cohen, S. Garfinkel and B. Schatz, Extending the Advanced Forensic Format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow, *Digital Investigation*, vol. 6(S), pp. 57–68, 2009.

[8] A. Flaglien, Cross-Computer Malware Detection in Digital Forensics, M.Sc. Thesis, Information Security Program, Faculty of Computer Science and Media Technology, Gjovik University College, Gjovik, Norway, 2010.

[9] A. Flaglien, A. Mallasvik, M. Mustorp and A. Arnes, Storage and exchange formats for digital evidence, presented at the *NISK Conference*, 2010.

[10] S. Garfinkel, Forensic feature extraction and cross-drive analysis, *Digital Investigation*, vol. 3(S), pp. 71–81, 2006.

[11] S. Garfinkel, Automating disk forensic processing with SleuthKit, XML and Python, *Proceedings of the Fourth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, pp. 73–84, 2009.

[12] L. Getoor, Link mining: A new data mining challenge, *ACM SIGKDD Explorations*, vol. 5(1), pp. 84–89, 2003.

[13] L. Getoor and C. Diehl, Link mining: A survey, *ACM SIGKDD Explorations*, vol. 7(2), pp. 3–12, 2005.

[14] P. Gladyshev, Formalizing Event Reconstruction in Digital Investigations, Ph.D. Dissertation, Department of Computer Science, University College Dublin, Dublin, Ireland, 2004.

[15] G. Gu, R. Perdisci, J. Zhang and W. Lee, BotMiner: Clustering analysis of network traffic for protocol- and structure-independent botnet detection, *Proceedings of the Seventeenth USENIX Security Symposium*, pp. 139–154, 2008.

[16] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten, The WEKA data mining software: An update, *ACM SIGKDD Explorations*, vol. 11(1), pp. 10–18, 2009.

[17] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco, California, 2006.

[18] D. Hand, H. Mannila and P. Smyth, *Principles of Data Mining*, MIT Press, Cambridge, Massachusetts, 2001.

[19] S. Hoffman, China hackers launch cyber attack on India, Dalai Lama, *CRN* (www.crn.com/security/224201581), April 6, 2010.

[20] T. Khabaza, Hard Hats for Data Miners: Myths and Pitfalls of Data Mining, White Paper, SPSS, Zurich, Switzerland, 2005.

[21] J. Mena, *Investigative Data Mining for Security and Criminal Detection*, Elsevier Science, Burlington, Massachusetts, 2003.

[22] E. Messmer, The botnet world is booming, *Network World*, July 9, 2009.

[23] National Institute of Standards and Technology, National Software Reference Library, Gaithersburg, Maryland (www.nsrl.nist.gov).

[24] G. Richard and V. Roussev, Next-generation digital forensics, *Communications of the ACM*, vol. 49(2), pp. 76–80, 2006.

[25] C. Schiller, J. Binkley, D. Harley, G. Evron, T. Bradley, C. Willems and M. Cross, *Botnets: The Killer Web App*, Syngress, Rockland, Massachusetts, 2007.

[26] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, San Diego, California, 2006.

[27] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, San Francisco, California, 2005.

[28] X. Wu and V. Kumar (Eds.), *The Top Ten Algorithms in Data Mining*, Chapman and Hall/CRC, Boca Raton, Florida, 2009.

[29] Y. Zeng, X. Hu and K. Shin, Detection of botnets using combined host- and network-level information, *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 291–300, 2010.