4-2010

Developing Cyberspace Data Understanding Using CRISP-DM for Host-based IDS Feature Mining

Joseph R. Erskine
Air Force Institute of Technology

Gilbert L. Peterson
Air Force Institute of Technology

Barry E. Mullins
Air Force Institute of Technology

Michael R. Grimaila
Air Force Institute of Technology

Recommended Citation

Erskine, J. R., Peterson, G. L., Mullins, B. E., & Grimaila, M. R. (2010). Developing cyberspace data understanding: using CRISP-DM for host-based IDS feature mining. Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research, 1–4. https://doi.org/10.1145/1852666.1852751

Developing Cyberspace Data Understanding: Using CRISP-DM for Host-based IDS Feature Mining [*] [†]

Joseph R. Erskine
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, Ohio
joseph.erskine@afit.edu

Gilbert L. Peterson
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, Ohio
gilbert.peterson@afit.edu

Barry E. Mullins
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, Ohio
barry.mullins@afit.edu

Michael R. Grimaila
Air Force Institute of Technology
2950 Hobson Way
Wright-Patterson AFB, Ohio
michael.grimaila@afit.edu

ABSTRACT

Current intrusion detection systems (IDS) generate a large number of specific alerts, but typically do not provide actionable information. Compounding this problem is the fact that many alerts are false positive alerts. This paper applies the Cross Industry Standard Process for Data Mining (CRISP-DM) to develop an understanding of a host environment under attack. Data is generated by launching scans and exploits at a machine outfitted with a set of host-based forensic data collectors. Through knowledge discovery, features are selected to project human understanding of the attack process into the IDS model. By discovering relationships between the data collected and controlled events, false positive alerts were reduced by over 91% when compared to a leading open source IDS. This method of searching for hidden forensic evidence relationships enhances understanding of novel attacks and vulnerabilities, bolstering ones ability to defend the cyberspace domain. The methodology presented can be used to further host-based intrusion detection research.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Neural Networks

General Terms

Theory

Keywords

data mining, intrusion detection

[*]This work is sponsored by the Air Force Office of Scientific Research (AFOSR/NL), Dr. Robert Herklotz, Program Manager.

[†]The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government.

1. INTRODUCTION

Two problems which impact the usability of intrusion detection systems are 1) a high incidence of false positive alerts, and 2) alerts which do not mirror human understanding. A false positive alert is an erroneous warning of normal activity as malicious activity. Alerts may be very specific, but do not provide actionable information to the network security analyst. The analyst must filter through false and uninformative alerts in order to find those which pertain to truly malicious activity. This filtering delays network defenders from taking actions to secure the network before more widespread damage can occur. In order to ease the burden on system security personnel, a methodology for reducing the incidence of IDS false positive alerts, while correctly identifying malicious events, is required.

This paper exercises a methodology to expand cyberspace data understanding for the purpose of improving threat detection accuracy. By identifying relevant features from a set of live response digital forensic tools used as sensors, this methodology identifies not only if a system is under attack or not, but also indicates what stage of an attack is occurring. Experimentation shows that this methodology drastically reduces false positive alerts, while providing more actionable alerts of true incidents.

2. BACKGROUND

An overview of the cyber attack process, intrusion detection systems, CRISP-DM and artificial neural networks serves as

background information leading up to the employed methodology.

2.1 Attack Process

Cyber threats have a goal to disrupt, degrade, or deny the confidentiality, integrity, or availability of a computer system. Threats come in many forms, from malware such as viruses, spyware and rootkits; hacking frameworks such as Metasploit [10], used to develop or launch pre-built malicious payloads at vulnerable target machines, are all-too-easily accessible by anyone with Internet access. McClure, et al. [10] present the stages of an attack process which starts with footprinting, scanning and enumerating activity. Depending upon the malicious user's goal(s), an attack will progress through other stages such as exploits to gain access, escalate privilege, pilfer data, plant back doors, and/or initiate a denial of service. Each action, normal or malicious, taken on a computer system leaves forensic traces. Forensic traces come in many forms, among these are resource (e.g., CPU, network, memory, disk) utilization deltas, state changes, series of system calls, and transaction logs of events. These forensic traces serve as potential data sources for identifying malicious events.

2.2 Intrusion Detection Systems

Intrusion detection systems (IDS) [6] play a major role in combating these threats. Traditional network IDS (NIDS) are studied at great length [1] [4] and have the ability to detect network protocol-based attacks. However, NIDS have no insight to a host's internal environment (memory, processes, system calls, file I/O). Host-based IDS (HIDS) [7] are used to detect malicious activity on a computer, and are run from within the environment being monitored. However, IDS systems typically focus on a single source of forensic data (e.g., network packets or series of system calls), and ignore other data sources which may be rich with additional evidence of malicious activity.

2.3 CRISP-DM

The CRoss Industry Standard Practice for Data Mining (CRISP-DM) [2] outlines a six-phase cycle for data mining projects, as shown in Figure 1. CRISP-DM is intended to be industry, tool and application independent. The goal of CRISP-DM is to provide organizations an understanding of the data mining process and provide a road map to follow while planning and carrying out a data mining project. The phases are business understanding, data understanding, data preparation, modeling, evaluation and deployment. The second phase, data understanding, "starts with initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, discover first insights into the data or to detect interesting subsets to form hypotheses for hidden information" [2]. This is the stage where the most work to identify relevant features occurs. Preparation involves transforming raw data into a dataset that can be used during the modeling stage. The modeling stage consists of implementing varied modeling techniques, such as using an artificial neural network [5]. Evaluation consists of analyzing how well the model meets the objectives, and is the final stage before deploying the model for use, and starting the whole cycle over again (as desired).

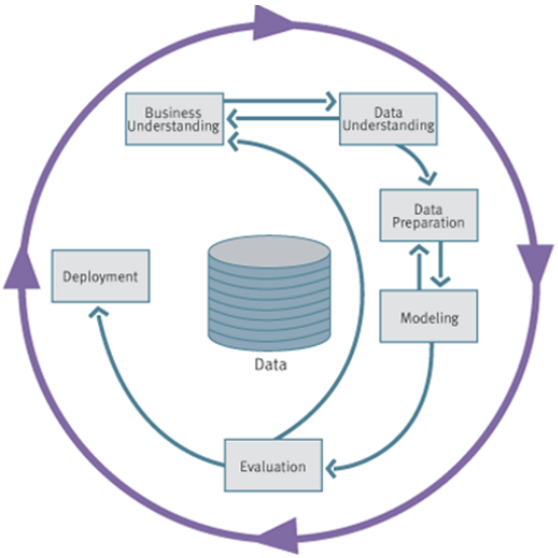

Figure 1: The Cross-Industry Standard Process for Data Mining is a cyclic process consisting of six-phases [2].

3. METHODOLOGY

In order to identify the forensic data which best indicates various stages of an attack on a host, CRISP-DM's data understanding, preparation, modeling and evaluation steps are used to analyze the forensic tool output for relevance in detecting malicious host activity.

3.1 Collection Environment

In order to facilitate experimental forensic collections, an experimental environment is established. The environment consists of a "black hat" machine and a target machine hosted on a common network which provides Internet access.

The black hat machine is a workstation outfitted with a number of cyber attack tools for scanning and exploiting the target machine.

The target is a Windows XP SP2 virtual machine (VM) outfitted with live forensic tools which monitor network, process and file activity. Two transactional sensors, which capture each transaction as it occurs, are employed – tshark.exe [3] for monitoring network packets sent to/from the host, and Event Tracing for Windows (ETW) [13] for monitoring process activity on the host. Additionally, four snapshot sensors, which capture a snapshot of a portion of the system's state in two-second intervals, are employed – SysInternals' [12] pslist.exe, listdlls.exe, tcpvcon.exe and logonsessions.exe. The target is implemented as a virtual machine (VM) to facilitate resetting the machine to a baseline configuration between data collections.

3.2 CRISP-DM: Data Understanding

During experiments, the target runs through scripted *normal* scenarios, such as using office automation products, an email client and a web browser. Additionally, the black hat runs through scripted *scanning* and *exploit* scenarios, such as using Nessus to scan the target and launching malicious

payloads at the target with Metasploit.

Collections are performed in three stages: *normal*, *scanning*, and *exploit* collections, generating three datasets to analyze. This helps to keep collection sizes manageable, and to provide distinct datasets based on the purpose for the collection. For each collection, the target machine is set to its baseline configuration, and the target machine's sensors are started. The *normal* collection generates a set of data which consists of typical user activity, devoid of scanning or other malicious activity. The *scanning* collection includes data which consists of the black hat machine performing scanning and service enumeration events. The *exploit* collection generates a set of data which consists of attempts to gain access, elevate privileges, plant back doors and to cover tracks. The activities from the three collections are logged and recorded for later data understanding analysis and future development.

Each dataset consists of outputs from the six sensors listed in Section 3.1, with potentially thousands of files to parse through. Each tool generates specific forensic data features. Each feature generated is initially considered a candidate for providing separation between normal and malicious activity, in order to identify as many effective features as possible.

3.3 CRISP-DM: Preparation

To facilitate data analysis, a framework for parsing the sensor data in a consistent and repeatable way is developed in Java. Since each sensor's output is specific, a distinct parser exists for each sensor in the framework. Parsing data from the selected snapshot sensors is relatively straightforward, as the outputs consist of structured text. However, the two selected transactional parsers generate binary files (ETL and PCAP files), which are translated to text using tracerpt.exe for ETL data, and jNetPcap [8] for network packets.

Data alignment is required in order to build a forensically diverse observation which includes data from each of the chosen sensors. Three elements are used to align data between sensors: Process ID (PID), Local Port, and Forensic Time Window (FTW). First, the PID associates observations from each of the SysInternals sensors to the ETW sensor. Local port associates network traffic from tshark to tcpvcon (and thus the rest of the sensors via PID). Another data alignment consideration is timing. Because snapshot sensors are executed one after another, there are discrepancies in the collection times for the "current" snapshot of a process. This is addressed using FTWs, a two-second time frame by which a process' or port's activity can be summarized. Shilland [14] takes this approach to account for variations in code speed on a MANET (MANET) IDS. Since collections are performed on one machine using a single computer's clock to track time, it is assumed time synchronization issues which arise from distributed systems [9] are not an issue.

With this in mind, features are identified using one of two methods: "Set distinction" and "Abnormal measures."

3.3.1 Set Distinction

The first method for identifying features is through set distinction. Datasets of distinct values occurring within a candidate feature are built using Transact SQL (T-SQL); one set is generated for each class of data to classify. From this, more T-SQL statements are executed to identify which values only occur in one collection or another. Results from this process are then manually examined to determine if the presence of a particular value is a matter of coincidence (actually part of normal operations) or appears correlated to a specific event or set of events. Items which appear correlated to a specific event are researched in literature to determine if the value has significant meaning to the event. From this, a decision is made to include the discovery of this value as a feature for HIDS or not.

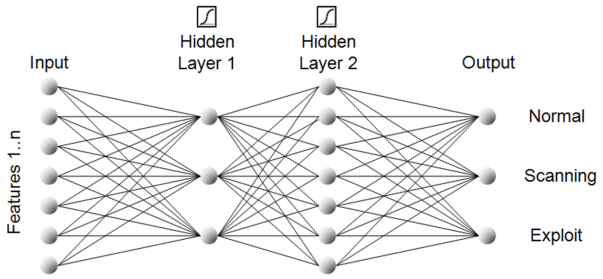

Figure 2: The ANN developed uses an input layer with 7 neurons, corresponding to the features selected during data understanding, two hidden layers of neurons with 3 and 7 neurons each, respectively, and an output layer of 3 neurons, corresponding to the three types of activity classified. The ANN learn rate is set to 0.01, implements the tansig [5] activation function, and has a goal error rate of 0.0001 or better.

3.3.2 Abnormal Measures

For features which rely on measures (e.g., counts of event x or Δ measures), an analysis of the distribution of the datapoints is performed. To do this, sample statistics are taken for each observed variable, namely its sample mean, standard deviation, skewness, and kurtosis. These measures help to determine each variable's potential value toward separating classes of data.

3.4 CRISP-DM: Modeling and Evaluation

An artificial neural network (ANN), depicted in Figure 2, is used to model the data set of selected features. Each observation is formatted appropriately for use with the MATLAB Neural Network Toolbox [5], to include the observation's true classification label and a vector of the selected features. The input data is split into two groups, populated via Monte Carlo sampling [11]: two-thirds for training and one-third for testing. The ANN is trained via back propagation, and consists of an input layer with 7 neurons, corresponding to the features selected during data understanding, two hidden layers of neurons with 3 and 7 neurons each, respectively, and an output layer of 3 neurons, corresponding to the three types of activity classified.

Once the ANN training phase is completed, the ANN is used to classify the testing data set.

4. RESULTS

Collection analysis identified seven specific items which correlate well with the events occurring at the time and pro-

Table 1: Comparison between Snort and ANN Ruleset false positive and false negative alerts.

Dataset	Rule Set	True Positive Alerts	False Positive Alerts	False Negative Alerts
Normal	Snort	0	285	0
	ANN	0	3	0
Scanning	Snort	548	4	9
	ANN	70	4	11
Exploit	Snort	57	0	6
	ANN	156	3	5
Total	Snort	605	289	15
	ANN	226	10	16

vided discrimination between normal and attack. Each feature is discovered independently through either set distinction or abnormal measures analysis. Six of the seven features are attributed to File IO activity, the seventh is a measure of port activity. Admittedly, some of these features may not perform well in environments which differ wildly from the environment used in this research.

1. *High count of local ports with activity.*
2. *File IO activity involving a /mailslot/.*
3. *File IO activity involving /mailslot/Nessus.*
4. *File IO activity involving /mup/.*
5. *File IO activity involving net.exe*
6. *Authentication File Dump Feature*
7. *Remote as System feature*

Overall, as summarized in Table 1, the methodology results in just 10 false positives and 16 false negatives, a 91.5% reduction when compared to Snort IDS' 289 false positives and 15 false negatives. Additionally, six events involving malicious remote shell connections were detected through the features discovered via data understanding which were not detected by Snort, and three times as many true positive alerts for exploit activity were generated by the data understanding ANN.

With regard to scanning, it is noted that Snort reported significantly more true positive alerts than the ANN method. Conversely, the ANN method reported significantly more true positive exploit activity alerts. As a NIDS, Snort signals an alert for each packet meeting a specific signature, such as A xmas tree scan or syn flood. The focus of this work is not to duplicate all the alerts that another system can create, but to validate the use of CRISP-DM to perform forensic data feature mining in order to improve the accuracy of an IDS. While the numbers appear skewed, a better comparison is between false positive and false negative alerts, as this is where one tool or the other misidentified normal activity as scanning or exploitation activity, or vice versa.

5. CONCLUSION & FUTURE WORK

Based on the results obtained, this research validates the use of CRISP-DM toward mining for relevant HIDS features. The HIDS features discovered yielded not only an improved detection of malicious activity, but also greatly reduced the incidence of false positive alerts when compared to Snort IDS.

As collections were done in a controlled environment; future efforts may focus on testing the effectiveness of identified features in larger, more operationally realistic network environments. Different attacks and attack tools could be used to further expand and validate the set of identified features deemed relevant to detecting malicious activity. Additionally, more or different sensors can be incorporated for monitoring other forensic aspects of the target system under attack. Such sensors include monitoring system calls and/or application program interface (API) calls and other inter-process communication.

6. REFERENCES

[1] S. Axelsson. Intrusion detection systems: A survey and taxonomy. *Depart. of Computer Engineering, Chalmers University, Tech. Rep*, pages 99–15, 2000.

[2] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth. CRISP-DM 1.0: Step-by-step data mining guide. *SPSS inc*, 78, 2000.

[3] G. Combs et al. Wireshark. *Web page: http://www.wireshark. org/last modified*, pages 12–02, 2007.

[4] H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion-detection systems. *Comput. Networks*, 31(8):805–822, 1999.

[5] H. Demuth and H. M. Beale, M. The neural network toolbox 6 for matlab user's guide. 2009.

[6] D. Denning. An intrusion-detection model. *IEEE Transactions on software engineering*, 13(2):222–232, 1987.

[7] S. Forrest, S. Hofmeyr, A. Somayaji, T. Longstaff, et al. A sense of self for unix processes. In *IEEE Symposium on Security and Privacy*, pages 120–128. IEEE COMPUTER SOCIETY, 1996.

[8] jNetpcap website. http://www.jnetpcap.org. 2009.

[9] L. Lamport. Time, clocks, and the ordering of events in a distributed system. 1978.

[10] S. McClure, J. Scambray, and G. Kurtz. *Hacking Exposed 6: Network Security Secrets & Solutions*. McGraw-Hill Osborne Media, 2009.

[11] S. Russell and P. Norvig. Artificial intelligence: a modern approach. *New Jersey*, 1995.

[12] M. Russinovich. Microsoft sysinternals suite.

[13] M. Russinovich and D. Solomon. *Microsoft Windows Internals*. Microsoft Press, 2005.

[14] G. Shilland and U. Major. Host-Based Multivariate Statistical Computer Operating Process Anomaly Intrusion Detection System (PAIDS), 2009.