Phase-Space Detection of Cyber Events

Jarilyn M. Hernández
West Virginia University
Lane Department CS & EE
P.O. BOX 6109
Morgantown, WV 26506
(865)-574-5480
jhernan7@mix.wvu.edu

Aaron Ferber
Oak Ridge National Laboratory
P.O. BOX 2008 MS6025
Oak Ridge, TN 37831
(865)-804-5161
ferberae@ornl.gov

Stacy Prowell
Oak Ridge National Laboratory
P.O. BOX 2008 MS 6418
Oak Ridge, TN 37831
(865)-241-8874
prowellsj@ornl.gov

Lee Hively
Retired from Oak Ridge National Laboratory
Chief Scientist, Gradient Dynamics LLC
(865)-376-2608
lee.m.hively@hughes.net

ABSTRACT

Energy Delivery Systems (EDS) are a network of processes that produce, transfer and distribute energy. EDS are increasingly dependent on networked computing assets, as are many Industrial Control Systems. Consequently, cyber-attacks pose a real and pertinent threat, as evidenced by *Stuxnet*, *Shamoon* and *Dragonfly*. Hence, there is a critical need for novel methods to detect, prevent, and mitigate effects of such attacks. To detect cyber-attacks in EDS, we developed a framework for gathering and analyzing timing data that involves establishing a baseline execution profile and then capturing the effect of perturbations in the state from injecting various malware. The data analysis was based on nonlinear dynamics and graph theory to improve detection of anomalous events in cyber applications. The goal was the extraction of changing dynamics or anomalous activity in the underlying computer system. Takens' theorem in nonlinear dynamics allows reconstruction of topologically invariant, time-delay-embedding states from the computer data in a sufficiently high-dimensional space. The resultant dynamical states were nodes, and the state-to-state transitions were links in a mathematical graph. Alternatively, sequential tabulation of executing instructions provides the nodes with corresponding instruction-to-instruction links. Graph theorems guarantee graph-invariant measures to quantify the dynamical changes in the running applications. Results showed a successful detection of cyber events.

Categories and Subject Descriptors

J.2 [**Computer Applications**] Physical Sciences and Engineering – *Engineering* K.6.5 [**Management of Computing and Information Systems**]: Security and Protection – *invasive software, unauthorized access.*

General Terms

Measurement, Performance, Design, Reliability, Experimentation, Security.

Keywords

Energy Delivery Systems, cyber anomaly detection, phase-space analysis, graph theory, malware, rootkits, cyber-attacks.

CISR '15, April 07 - 09, 2015, Oak Ridge, TN, USA © 2015 ACM. ISBN 978-1-4503-3345-0/15/04…$15.00

DOI: http://dx.doi.org/10.1145/2746266.2746279

1. INTRODUCTION

Energy Delivery Systems (EDS) and their embedded software components are critical to the reliable operation of the United States' critical infrastructure. A successful attack against these distributed, interoperating, mostly privately-managed systems would be devastating to the economy of the United States.

A recent study conducted by B2B International and Kaspersky lab revealed that damages from a successful targeted cyber-attack against a single company average $2.4 million [1]. These numbers have been dwarfed by recent exploits against nationwide businesses, each with damages estimated in hundreds of millions of dollars. These latter cyber-attacks against nationwide businesses resemble the corporate structure, economic forces, vendor and customer relations, and nationwide distribution that apply also to EDS.

Two recent examples of cyber-attacks that have targeted critical infrastructures are Shamoon [2, 3] and Dragonfly [4, 5]. Shamoon targeted a national oil company in Saudi Arabia called Aramco. It spread through Aramco's corporate network wiping and overwriting computer files [6, 7, 8]. Approximately three-quarters of the corporate PCs were affected by this attack, and the company was forced to stop oil production for a week before services could be restored [8]. Dragonfly targeted energy grid operators, electricity generation firms, petroleum pipeline operators, and energy industry industrial equipment providers. The majority of its victims were located in the United States, Spain, France, Italy, Germany, and Poland [4].

These examples show that there are motivated and well-financed communities around the world that are targeting national infrastructure with near immunity. EDS and their embedded software components are being increasingly exposed to cyber-attacks. The existing cyber security protection systems are often too focused on limited samples, overwhelmed by a deluge of data, and complex. The development of new and robust methods to detect cyber-attacks in these systems is imperative.

In this paper, we present a prototype implementation for detection of intrusions by measuring the precise execution time of a collection of events in EDS. Specifically, we developed a framework for gathering and analyzing timing data that involves establishing a baseline execution profile and then capturing the effect of perturbations in the state from injecting various malware.

In this paper, Section 2 describes the technical approach for phase-space analysis of time serial, process-indicative data. Section 3 describes the experimental design. Section 4 presents the results that we obtained from our experiments. Section 5 presents how the client works, how the server works, and how communications between the client and the server are established. Section 6 includes our conclusions and future research goals.

2. RELATED WORK

Time-serial measurements of power consumption have been used to extract cyber dynamics by a nonlinear dynamical approach [9-10]. The method creates dynamical states by the time-delay embedding theorem [11]. The analysis found strong evidence for a low-dimensional attractor in simple programs, as well as showing the first experimental evidence of chaos on a real computer [9]. Differential power analysis was used for adversarial exploitation of cryptographic hardware [12-14].

Given historical success for using theorem-based, data-driven analysis techniques in biomedical and industrial applications [15-16], we postulated that side-channel characterization from non-invasive sensors may provide indicators for predicting failures in physical devices and detecting execution of anomalous software.

The garbage-in-garbage-out syndrome was avoided by rejecting inadequate data that fail any of the following tests [17]: proper number of data points; lack of signal variation; saturation at high or low limits; consistent amplitude across datasets; adequate sampling rate; excessive periodic content; and excessive noise. This analysis avoids meaningless results from poor-quality data.

Takens' theorem [11] gives a smooth, non-intersecting dynamical reconstruction in a sufficiently high dimensional space by a time-delay embedding. See [18] for a review. The symbolized data are converted into unique states by a time-delay-embedding vector, y_i:

$$y_i = [s_i, s_{i+L}, \ldots, s_{i+(d-1)L}] \tag{1}$$

This theorem converts time-serial dynamics to topology. Namely, Takens' theorem says that the y_i-states are diffeomorphic to the underlying dynamics, as a way to capture topology (connectivity and directivity). The time-delay lag is L, which must not be too small (making s_i and s_{i+L} indistinguishable) or too large (making s_i and s_{i+L} independent by long-time unpredictability). The embedding dimension is d, which must be sufficiently large to capture the dynamics, but not too large to avoid over-fitting.

Another provable component involves time-delay states from Eq. (1) as nodes. The process flow, $y_i \rightarrow y_{i+M}$, forms state-to state links. The set of nodes and links form a mathematical "graph." Graph theorems guarantee novel, topologically-invariant measures from this graph, meaning that the measures are independent of labels for the nodes and links [19]. This graph is "directed" or a "digraph," meaning that the flow is from one state (y_i) to another (y_{i+M}), but not necessarily in the reverse direction.

This analysis uses four graph-invariant features to measure dissimilarity between cutsets: (1) nodes in digraph A but not in B; (2) nodes in B but not in A; (3) links in A but not in B; and (4) links in B but not in A. These measures sum the absolute value of differences, providing better discrimination then traditional nonlinear measures, which use a difference of averages. The measure is normalized to the number of nodes (links) in A (for A not in B) or in B (for B not in A). These measures form the feature vector, V, to classify the data as giving detection of event or not. The analysis obtains a vector of mean dissimilarities, $\underline{V}$, and corresponding standard deviations (σ) by comparison among the $B(B-1)/2$ unique combinations of the B base-case graphs. Each subsequent test-case graph is then compared to each of the B base-case graphs to obtain an average dissimilarity vector, v. An event indication is several successive instances (K) at a threshold (U_T) for each feature, $U(V) = |v - \underline{V}|/\sigma$, for J features.

Statistical validation of event detection (or forewarning) requires objective measures of success. One measure is the number of true positives (TP) out of the known event datasets (Ev), to yield the true positive rate (specificity) of TP/Ev. A second measure was the number of true negatives (TN) from known non-event (NEv) datasets. The true negative rate is TN/NEv (sensitivity). Consequently, minimizing the distance from ideal (D = prediction distance) is an appropriate objective function for any event type:

$$D = \{[1 - (TP/Ev)]^2 + [1 - (TN/NEv)^2]\}^{1/2} \tag{2}$$

Clearly, excessive false positives (inverse of a true negative) will cause real alarms to be ignored, and needlessly expend responder resources. False negatives (inverse of a true positive) provide no forewarning of events, which could then prove disastrous.

The phase-space analysis technique has shown historical success for forewarning of biomedical events (e.g., epileptic seizures [20], sepsis, and breathing difficulty [21]), as well as forewarning of equipment failures (e.g., motors, gears, spindles [16], [20-21]).

Also, in the past other researchers have attempted to analyze malicious software using a kernel level analysis. The work in [22] presented a technique that was based on static analysis to identify instruction sequences that are indication of rootkits. This technique exploits binary analysis to ascertain, at load time, if a module's behavior resembles the behavior of a rootkit. Similarly, the work in [23] presented a framework that uses the information in kernel structures of a process to do runtime analysis of the behavior of an executing program on Linux platform by applying machine learning techniques. The difference between previous work and our approach is that we are not only combining some of the key concepts from [22] and [23], but we are adding an analysis based on Taken's theorem.

In addition, the work presented in [24] introduced, WattsUpDoc, a behavior-monitoring system that detects malware on embedded medical devices and on a SCADA device by measuring the power consumption of these devices. Specifically, their work used machine learning techniques to model permissible behavior and detects deviations. The difference between the approach of [24] and our approach is that we are using timing data to detect anomalies on the system. Also, while they are using machine learning techniques to detect anomalies, our data analysis was based on nonlinear dynamics and graph theory.

Our novel work seeks detection of anomalies in cyber behavior from the execution times of various system calls. System calls provide an interface between the user application and the Linux kernel. They take the burden and risk of accessing underlying resources out of the hands of application programmers. At the same time they are a prime target for rootkits, because they access privileged data and execution contexts. They also provide a top level view of the kernel, meaning many underlying kernel functions are called through system calls. For these reasons they are ideal for detecting function timing perturbations in computer systems.

3. EXPERIMENTAL DESIGN

The objective of our experiments was to gather timing data in a clean computer system with no malware, and subsequently to gather data from an infected system. The goal of the data analysis was to establish a difference between the nominal and anomalous behavior. We began our experiments with a clean installation of Ubuntu 12.04 (Desktop i386), and gathered nominal data. Then, we infected the machine with a rootkit called *kBeast*, which replaces function pointers to system calls and changes the tcp4 process table (among other things).

For our experiments, we chose five applications: *Libre Office Writer* (Version 3.5.7.2), *Document Viewer* (Version 3.4.0), *Firefox* (Version 18.0.2), *Calculator* (Version 6.4.1.1), and *Mines* (Version 3.4.1). These applications were selected because they were already installed by default on the operating system and also because these applications are popular among users.

Each application was executed separately in two different scenarios: nominal behavior (no malware running on the system) and anomalous behavior (rootkit running on the system). The five applications with two scenarios produced 10 different datasets. During each computational experiment, we monitored the execution time for two system calls: *sys_open* and *sys_read*.

We collected 1 million points per system call for each experiment, resulting in 2 million points for each nominal/rootkit dataset. The average time for the server to collect the data was approximately 18 minutes. While the server was gathering the data, we used the applications as a regular user.

4. RESULTS

For the event-detection analysis, we used the execution time for each call to *open* or *read*. The nominal data had 1 million data points, and event data had an additional one million data points. These data were concatenated into a single dataset (2 million data points). The phase-space dissimilarity measures were calculated without artifact filtering, using the following parameters: number of symbols, S=10; time-delay lag, L=1; inter-symbol lag, M=1; number of basecase cutsets, B=10; and number of phase-space dimensions, d=2. The data was divided into analysis intervals (cutsets); the number of points in each cutset was N=50,000, yielding 20 nominal cutsets and 20 event cutsets.

Figure 1 shows the results for no background application with execution times for each call to OPEN as the observable data. The top subplot shows the results for the first dissimilarity measure (nodes in graph A, but not in graph B). The blue curve in Figure 1 displays the dissimilarity measures versus cutset number; these dissimilarities were highly variable and have no obvious pattern. The red curve shows discrete-valued features, which range in value from 1 to 4. These features use one of four Matlab$^{\text{TM}}$ functions: *ROUND*, *FLOOR*, *CEIL*, or *FIX*. A numerical offset was applied to the dissimilarity measure, chosen from all of the possible values of this dissimilarity measure. The feature value was linearly offset to have a value ≥1. The black star (*****) indicates the event detection as five successive occurrences of the feature at a value of 2. The choice of feature function and offset was based on an exhaustive numerical search for the earliest unique event detection after cutset #20 (end of the nominal data).

The results of our experiments showed that cyber events can be successfully detected through the analysis of timing data. These results are significant since to the best of our knowledge this is the first time in which Taken's theorem is used to detect anomalies through monitoring and analyzing the execution time of system calls to detect cyber anomalies.

The same analysis was applied to each of the dissimilarity measures for each of the computation experiments, as Figure 1 depicts [25]. Earlier event detection might be obtained by optimizing the analysis parameters, which was not attempted for this initial demonstration.

5. PROTOTYPE IMPLEMENTATION

This section explains how the client and server architecture works. It also explains how the communication is established between the client and the server.

5.1 Execution Flow for the Client

The client is a Loadable Kernel Module (LKM). It gathers timing data from important kernel functions and data structures, and sends the analysis of the data to the server. Data is sent as a JSON (JavaScript Object Notification) file.

CUTSE

Figure 1: (No App Open) Dissimilarity Measures (blue), features (red), and event detection (*)

The client begins by connecting to a server via TCP/IP, using AES 256-bit symmetric encryption with an electronic codebook block cypher. The client continuously gathers timing data in the form of cutsets. After a cutset is gathered, it is sent to a user-space process for analysis. The analysis returns data as dissimilarity measures. The client then creates a JSON file using these measures, and then it will connect to the server. The JSON file is sent over this connection with a 256-bit AES encryption. Once the JSON file is sent, the client disconnects from the server and waits a set amount of time before gathering more timing data.

The LKM works by creating a kernel process file, which accepts commands and returns data via read/write access. The commands specify which system calls to monitor, what timing devices to use, and how many data elements are needed. Once a command has been issued, it is executed within the kernel and the data is stored with the process file. The next time the process file is read, the data will be copied to user space. Specifically, the LKM handles its own thread. This thread will continuously check the process file for new commands. Once a new command it's received, it is parsed and executed. The execution consists of copying the system calls into kernel memory and timing the execution of copied calls. The challenge with the proposed framework is implementing timing routines for a sufficient amount of critical system functions. Until such implementations are in place, a novel rootkit could operate outside the set of monitored functions.

5.2 Execution Flow for the Server

The server continuously accepts TCP/IP connections from any client that tries to connect to it. After the server has accepted a connection with a client, it creates a thread for the client and continues to wait for more clients to connect. Using threads allows for multiple clients to connect and send data asynchronously to the server. The thread then performs an analysis on the data contained in the JSON file. The client thread then connects to a MySQL database and inserts the data contained in the JSON file. When the thread finishes inserting data into the database, it waits for the client to disconnect.

In order to send a large amount of data over the TCP connection, the client must split the data into smaller parts and send each part to the server. For this reason, the server first receives the number of parts into which the message split. Then, the server expects to keep receiving data until the entire dataset has been collected. If at any time the server receives less than or equal to zero bytes of data from the client, it disconnects from the client and ends its thread.

6. CONCLUSIONS AND FUTURE WORK

This work demonstrated a novel approach for cyber event detection. We showed that the technical strength of the approach is a theorem-based, data-driven, phase-space analysis. After conducting our experiments, gathering and analyzing the data, we conclude that we can successfully detect cyber events. Our future work includes improvements to the client and server architecture. Specifically, we want to make our approach compatible with 64-bit Linux and develop a Windows version of this framework.

7. ACKNOWLEDGMENTS

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

8. REFERENCES

[1] H. Mackenzie, “Shamoon Malware and SCADA Security What are the impacts?”, 2012.

[2] “Shamoon”, http://en.wikipedia.org/wiki/Shamoon.

[3] Symantec Security Response, “Dragonfly: Western Energy Companies under Sabotage Threat”, 2014.

[4] H. Mackenzie, “Dragonfly Malware Targets ICS Systems”, 2014.

[5] Kaspersky Lab, “Targeted cyber-attacks cost companies up to $2.4 million in damages”, 2013.

[6] D. Walker, “Data-wiping Shamoon targeting Middle East energy sector”, 2012.

[7] BBC News, “Shamoon virus targets energy sector infrastructure”, 2012.

[8] N. Perlroth, “In Cyberattack on Saudi Firm, U.S. sees Iran firing back”, 2012.

[9] Takens, F., Detecting strange attractors in turbulence. In Rand, D.A. and Young L.S., *Dynamical Systems and Turbulence, Lecture Notes in Mathematics* 898: 366–381, Springer-Verlag, 1981.

[10] Alexander, Z., et al., Measurement and dynamical analysis of computer performance data. IDA 2010: pp. 18-29, 2010.

[11] Mytkowicz, T., et al., Computer systems are dynamical systems. *Chaos* 19: 033124, 2009.

[12] Alexander, Z., et al., Measurement and dynamical analysis of computer performance data. *IDA* 2010: pp. 18-29, 2010.

[13] Kocher, P., et al., Introduction to differential power analysis and related attacks. *Technical Report*, Cryptography Research Inc., 1998.

[14] Kocher, P., et al., Differential power analysis. In Proc. 19th Annual Int’l *Cryptology Conference on Advances in Cryptology*, pp. 388–397, Springer-Verlag, 1999.

[15] Mangard, S., Oswald, E., and Popp, T. Power analysis attacks: Revealing the secrets of smart cards. In *Advances in Information Security*, Springer-Verlag, 2007.

[16] Protopopescu, V. and Hively, L.M. Phase-space dissimilarity measures of nonlinear dynamics: Industrial and biomedical applications. *Recent Res. Devel. Physics*, 6, 649-688, 2005.

[17] Hively, L., M., Protopopescu, V.A., and Munro, N.B. Epilepsy forewarning via phase-space dissimilarity. *J. Clin. Physiol*. 22, pp. 402-409, 2005.

[18] Sakkalis, V., Review of advanced techniques for the estimation of brain connectivity measured with EEG/MEG, *Comput. Biol. & Medicine*, Vol. 41, pp.1110-1117, 2011.

[19] Bondy, J.A., and Murty, U.S.R, *Graph Theory, Springer*, ISBN 978-1-84628-969-9, 2008.

[20] Hively, L., et al., *Forewarning of Epileptic Events from Scalp EEG*, peer-reviewed proceedings paper for Biomedical Science and Engineering Conference at ORNL to be published.

[21] Protopopescu, V., and Hively, L., Phase-space dissimilarity measures of nonlinear dynamics: Industrial and biomedical applications, *Recent Res. Dev. Physics*, Vol. 6, pp. 649-688, 2005.

[22] Kruegel, C., et al., Detecting Kernel-Level Rootkits through Binary Analysis. *In Proc of the 20th Annual Computer Security Applications Conference*, Washington, DC, USA, pp. 91-100.

[23] Shahzad, F., et al., In-Execution Malware Detection Using Task Structures of Linux Processes. In Proc *of the IEEE International Conference on Communications*, Kyoto Japan, pp. 1-6.

[24] S. Clark, et al., “WattsUpDoc: Power sides channels to nonintrusively discover untargeted malware on embedded medical devices”, in proceedings of USENIX Workshop on Health Information Technologies, 2013.

[25] Hernández, J. et al., Beholder: Phase Space Detection of Cyber Events, *ORNL/TM-2013/294*, Oak Ridge National Laboratory, Oak Ridge, TN, 2013.