Developing a Power Measurement Framework for Cyber Defense

Jarilyn M. Hernández
West Virginia University
College of Engineering and Mineral Resources
PO Box 6109, Morgantown WV 26506
(787)-408-6637
jary.hernandez@gmail.com

Line Pouchard
Oak Ridge National Laboratory
1 Bethel Valley Road
P.O.BOX 2008 MS 6367
Oak Ridge, TN 37831
(865)-574-6125
pouchardlc@ornl.gov

Jeffrey McDonald
University of South Alabama
Shelby Hall
150 Jaguar Drive
Mobile, AL 36688
(251)-460-7555
jtmcdonald@southalabama.edu

ABSTRACT

Energy Delivery Systems (EDS) have become smarter by incorporating chips and data communication capabilities. As a result, they have become more vulnerable to cyber-attacks as well. As part of a systems health monitoring approach, we investigate whether cyber-events targeting EDS can be detected by monitoring component-level data such as temperature, voltage, power, and process indicative variables—collectively referred to as *component health indicators*. We report our experiences with developing a measurement framework for power consumption in different EDS components such as Cabinet Distribution Units (CDU), Power Distribution Units (PDU), and standard enterprise desktops. Our plan for gathering and analyzing power consumption data involves establishing a baseline execution profile and then capturing the effect of perturbations in the state from injecting various malware. As a contribution, we report on initial experiments with power measurement techniques and outline future work for evaluating components under normal and anomalous operating regimes.

Categories and Subject Descriptors

B.8.1 [**Hardware**]: Performance and reliability – *Reliability, Testing, and Fault-Tolerance.* 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, malware, rootkits, cyber-attacks, Power Distribution Unit, Cabinet Distribution Unit, Simple Network Management Protocol.

CSIIRW '12, October 30 - November 2, Oak Ridge, Tennessee, USA

1. INTRODUCTION

Millions of software-controlled, networked devices are being deployed as part of the next-generation EDS [1]. These systems and their embedded software are critical to the reliable operation of the North American power grid [2]. In our context, EDS are a network of processes that produce, transfer and distribute energy. The most common EDS examples are Supervisory Control and Data Acquisition (SCADA) and the Distributed Control Systems (DCS). EDS have become smarter by incorporating chips and data communication capabilities. As a result, they have become more vulnerable to cyber-attacks as well. The electrical power system has always been a high priority target for military and insurgents [3]. Attacking power plants, including substations and hydroelectric facilities is a normal part of guerilla warfare [3]. Cyber attack of EDS systems was first introduced in 2010 through the Stuxnet computer worm, the first to include a Programmable Logic Controller rootkit to industrial devices [4].

Collecting and analyzing machine health data is already of great importance in monitoring power consumption, improving resource management and scheduling, and refining application signatures for intrusion detection. Machine health data also offers new possibilities for detecting anomalous events, especially execution of malicious software, which would reduce vulnerabilities in EDS systems across the board. We describe the set-up of an experimental environment and data collection framework to evaluate possibilities for cyber event detection. We relate the technical details of our approach, the encountered challenges, and preliminary results. As a key contribution, we show the use of tools for monitoring and gathering health profiles in the context of anomaly detection, outside their traditional uses for enhancing EDS reliability and efficiency.

The paper is organized as follows: section 2 describes related work in hardware measurement and section 3 details our experimental approach for incorporating power measurement data in cyber event detection applications. Section 4 gives data collection results using software-based application programming interfaces (APIs) related to hardware-based sensors and section 5 describes our data collection activity with aggregate power monitoring devices such as CDUs and PDUs that provide Simple Network Management Protocol (SNMP) support. Section 6 includes our conclusions and future research goals.

2. RELATED WORK

In traditional applications, hardware measurements are collected and analyzed to understand the behavior of computer systems with regard to their time to failure (reliability) and for the purpose of monitoring power consumption (green computing or power

profiling). The reliability literature shows that hardware data collection methods are diverse and platform dependent [5]. In the absence of root-cause analysis, studies disagree on attributing failures to hardware, software or other causes [6]. Board, core or disk temperatures correlate to failures in some studies but not others [7,8]. Some systems collect hundreds of variables per node, analyze on the fly, and save only analysis results, thus preventing comprehensive forensics or reuse of their data [9,10]. Hsu and Poole in [11] detail state of the art in power measurement and classify hardware monitoring methods from node component level to facility. Currently, power consumption of high performance computer systems is closely monitored to develop power-aware job schedulers and reduce energy costs.

Cui *et al.* [12] report results related to power profiling and energy consumption in modern computers that bear on cyber event detection research. Their approach involves a uniquely built sensor device measuring individual component power. The method synchronizes measured power data with application code, demonstrating how program phases can be tracked. Their results show that power fluctuation is very rapid and demands fairly high sampling rates (50 KhZ) for fine grain analysis.

Buennemeyer *et al.* [13] propose a Battery Sensing Intrusion Protection System (B-SIPS) for mobile computers. The detection capability is scalable and compatible with existing commercial and open Intrusion Detection System (IDS). This implementation correlates device power consumption data with IEEE 802.11 Wifi and 802.15.1 Bluetooth communication. To detect the presence of malware they are using an unexplained increase in the current drawn from a device's battery.

When put in the context of the Stuxnet attack, Cui and Buennemeyer's research highlight the need to detect anomalies in EDS performance. Initially spread via Microsoft Windows, and targeting Siemens Industrial hardware and software, Stuxnet operates in two stages: first, it uploads configuration information about the Siemens system to a command-and-control server; and second, uses a rootkit on a target to alter its functions [4].

Our effort involves developing a power measurement framework for cyber event detection. The purpose is to determine if malware attacks can be detected from power consumption data such as voltage and current.

3. EXPERIMENTAL APPROACH

The goal of our research is to determine if cyber-events affecting EDS components can be detected through installed sensors and continual monitoring of temperature, voltage, power, and other component level indicators. The research effort is allocated into three phases: 1) baseline characterization; 2) perturbation experiments; and 3) algorithmic analysis. Our aim is to establish profiles for aggregate and component level measures that will enable us to investigate potential effects of perturbations caused by malicious software such as viruses, rootkits, Trojans, and backdoors. After data collection, we will employ anomaly detection and characterization algorithms to determine if the attacks are visible or not in component power profiles. In this paper we report initial results in configuring the data collection framework that will support the research phases. Our initial experiments use standard desktop personal computers for the testbed environment.

Our approach creates baseline profiles under two separate load regimes. The first regime involves monitoring and recording EDS component behavior with normal operating system and kernel level processes. The second regime involves execution of various algorithms and applications that stress CPU, memory, cache, and disk. Our EDS target uses a clean installation of 32-bit Windows 7 Enterprise. The clean installation assumes no malware is introduced through the operating system (OS) installation process.

Once we record baseline component health data, we will infect EDS components with functionally-profiled malware. The functional profile can be obtained by using custom-made rootkits where the behavior is known in advance. Our initial perturbation experiments will use rootkits, which are programs that operate typically with kernel level privileges and hide key indicators of their presence from the host OS. Rootkit behavior varies based on what goals they are designed to achieve. Some rootkits can install key-loggers or provide backdoors for later nefarious purposes; in general, they provide access for unauthorized users of a system [2].

Malware injection will allow us to create perturbation profiles consistent with the baseline profiles. While the infected EDS target is running, we will collect EDS component health data under the same load regimes as the baseline profiles: first, using normal OS processes, and second, using specific algorithms with particular performance characteristics. Statistical analysis follows data collection activities. The analysis methods will compare the baseline and perturbation profiles to determine if cyber-events or intrusion can be detected.

4. HARDWARE SENSOR APIs

Many modern processors now come with standard hardware-based sensors built into the motherboard that report system level information such as temperature, power, and processor statistics. For older EDS components, these sensors may not be present. Without the appropriate sensors, APIs that query the hardware for data will not work correctly. Discovering whether or not such sensors exist onboard can be a challenging task: we used several different tools to determine if sensors are present and our initial experiments utilized three standard API packages: *IPMItools*, *lm-sensors*, and *sensord*.

4.1 IPMItools

IPMItools is a utility for managing and configuring devices that support the Intelligent Platform Management Interface (IPMI). IPMI is a computer system interface used by system administrators to manage computer systems and monitor their operations. IPMI coordinates messages from on-board sensors and communicates Sensor Data Records (SDR) to the Sensor Data Repository [5]. Depending on the platform, IPMI essentially reports temperatures, voltages, and fan revolutions per milli-second (rpms).

We note that this tool attributes measurements to a particular temperature sensor or fan on a board [5]. For example *IPMItools* will return data such as *Temp1*, *Temp2*, *Temp3*, *Fan1*, etc. The tool also reports temperature data as integers in Celsius, making it less sensitive to capture rapidly rising trends and disallowing margins for corrective action [5]. *IPMItools* was installed in the Ubuntu OS environment. The command *sudo ipmitools sdr list* detects if a system is compatible with *IPMItools* and returns all the sensors embedded in the system. Sensor values at the moment of execution and the Sensor Data Record (SDR) list are also returned. . A sensor data record repository provides the properties of the sensors located on the board. This procedure was done in three standard enterprise desktops but no sensors were found. Therefore using standard enterprise desktops for gathering the power consumption data was discarded as we discovered that our PC testbeds were not compatible.

Figure 2. Results of *ipmitools SDR list* command.

We installed *IPMItools* in a cluster computing environment at Oak Ridge National Laboratory (ORNL) using the same command. Figure 2 shows the sensors that were detected in the cluster. The next section defines lm-sensors, why it was used, and what the obtained results were after using this tool.

4.2 lm-sensors

Lm-sensors is an open-source tool compatible with Linux distributions. It detects the sensors that are embedded in the motherboard of any machine. To install this tool in an Ubuntu machine, we use the command *sudo apt-get install lm-sensors*, and to execute the tool we use the command *sensor*s. Figure 3 illustrates typical results from the *sensors* command.

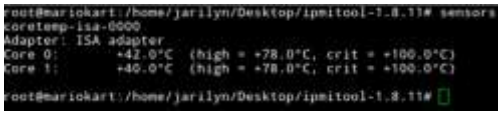

Figure 3. Results of *sensors* command.

As figure 3 illustrates, the temperature of the cores of the EDS component are displayed. Because these measurements only report precision at one decimal place and because the tool reports one variable (the CPU core temperature), we conclude that it is not useful for future cyber event detection experiments.

4.3 sensord

Sensord is a daemon that is part of the *lm-sensors* suite. This tool used a round-robin database (*RRDtool*) to monitor and plot the data in real-time. A round-robin database is a high performance data logging and graphing system for time series data. Figure 4 shows the results of the core temperature measurements from the same EDS component shown in figure 3. This tool allowed us to verify the format of data and characterize options for plotting. In the end, the format of the data was the same as *lm-sensors* and we deemed it insufficient for the collection framework.

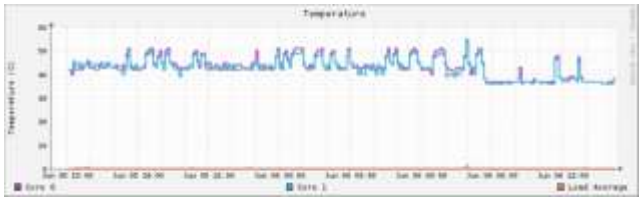

Figure 4. Typical *sensord* data results.

5. AGGREGATE POWER MEASURES

High performance computing centers use power monitoring devices that provide management features for operators. Because they are easily acquired and installed at a reasonable price, we evaluated whether such devices could detect cyber events in power measurement applications.

5.1 Power Distribution Units

A Power Distribution Unit (PDU) is a device with multiple outputs designed to distribute electric power. There are two major classes of PDU; the first one corresponds to the category of relatively higher-cost floor-mounted power distribution devices and the second class corresponds to smaller device.

In our experiments we used a rack PDU switched, a device from the first class. The router provides access to the collected data. However the minimum sampling rate was once per minute. Our experiment requires the option to modify the sampling rate. Therefore, using the PDU was not an option.

5.2 Cabinet Distribution Units

A cabinet distribution unit (CDU) is a device for power monitoring. CDUs typically provide Simple Network Management Protocol (SNMP) interfaces to monitor, track and manage servers and IT equipment and the cabinet infrastructure that house them. We used the Sentry CWG-8H1 Switched CDU to measure power consumption data of two experimental machines. The CDU provides per outlet power sensing (POPS) data such as power (in watts), current (in amps), and the voltage of the devices that are plugged into it. Configuring CDUs is simple: plug a monitored device into the CDU, connect the CDU via Ethernet to a network appliance, and connect a monitoring collection machine (PC) via Ethernet into the same network appliance. However, finding a monitoring tool that will support reasonable data sampling rates and provide data export capabilities is a more challenging task. By default the settings and data extracted from the CDU can be accessed by interfacing with default IP of the device (*192.168.1.254*). Communication between the monitoring PC and CDU depends on configuring the SNMP settings correctly. Correct configuration was achieved but SNMP collection tools (either open source, commercial, or custom) are still required.

5.3 SNMP Collection Tools

Simple Network Management Protocol (SNMP) is an application layer protocol used to manage network resources. This standardization gives network administrators the ability to monitor network performance [14]. There are hundreds of SNMP management tools that can interface with a CDU, however most of them support network administration and management tasks (not higher precision or higher frequency data collection). We evaluated three tools to see if they were applicable for our cyber defense power measurement framework: *Loriot Pro*, *Solarwinds Workspace Studio*, and *Sentry Power Manager* from *Server Technology*.

Loriot Pro met many of our requirements: it detects all CDU devices in the network and has built-in performance monitoring for connected machines. However, we were unable to extract the power consumption data from the CDU via the standard SNMP configuration options.

Solarwinds Workspace Studio allows monitoring CPU and memory usage for attached machines, but does not support power measurement. *Sentry Power Manager* is an SNMP management tool that runs on Linux installations. Using *Sentry Power Manager*, we obtained power consumption data with more than three decimal places from monitored devices, an improvement over other devices we tested. But the sampling rate was every 6 minutes, lower than that of other devices. The following section

explains the system set-up for monitoring and gathering the power consumption data.

5.4 Experimental Configuration

The network configuration involved 1) the experimental machines (for baseline and perturbation experiments); 2) a data collection machine; 3) a network switch; and 4) the CDU. The collection machine uses SNMP to extract data from the CDU and exports the data into a CSV file. Figure 7 illustrates our configured network. We used both Ubuntu and Windows for the data collection machines and 32-bit Windows 7 Enterprise for the experimental machines.

Figure 7. Data collection network

Figure 8 illustrates data collection from experimental machines connected to the CDU. Our collection configuration is ideal because it allows us to perform unobtrusive observation of machines in baseline operating regimes. It also allows perturbation observation using malware injection without the need for virtualization and without danger of infecting the collection machines and network.

Figure 8. CDU monitored power for experimental machines.

6. CONCLUSIONS and FUTURE WORK

A key objective of our research involves developing a power measurement framework that can support experiments related to anomaly detection for cyber events, particularly execution of malicious software. We setup a test network and evaluated efficacy of several power measurement techniques, including hardware-based sensor APIs and aggregate power measurement tools for CDUs.

Of all the tools tested, the only tool that allowed us to gather and export the power consumption data was the *Sentry Power Manager*. This tool allowed us to gather the data with more than three decimal places and supported data export in extensible markup language (XML) format. But the sampling rate supported through standard SNMP tools was insufficient for our purposes. Our initial goal was 200 samples per second using general purpose tools. Our ultimate goal will require specialized, component-level monitoring tools that can support higher sampling rates.

After running initial tests with both hardware sensor API suites and CDU/SNMP tools, we conclude the following:

- Custom SNMP data collection software is required to extract the power consumption data at an appropriate sampling rate from CDUs.
- Hardware-based sensor API suites require the collection machine and monitoring machine to be one in the same, causing the collection to affect overall power readings. Some motherboards do not have built-in sensors and some tools do not work with sensors in a universal manner.

Our future work will continue to evaluate power collection techniques to find the right balance of cost, availability, sampling rate, and precision.

7. REFERENCES

[1] Prowell, S., Pleszkoch, M., and Linger, R. Automated Vulnerability Detection for Compiled Smart Grid Software. In *2012 IEEE Innovative Smart Grid Technologies (ISGT)*.

[2] TenHoopen, P. *Rootkits*. Presentation to West Michigan Linux Users Group. http://www.wmlug.org/pdf/Rootkits.pdf, 2009.

[3] Lewis, J. The Electrical Grid as a Target for Cyber Attack. *Center for Strategic and International Studies*, March 2010.

[4] Thabet, A. *Stuxnet Malware Analysis Paper*.

[5] Pouchard, L., Dobson, J., and Poole, S. Collecting sensor data for high-performance computing: a case-study. In *Proc. of the Parallel and Distributed Parallel Processing Techniques and Applications*, 2010.

[6] Cappello, F. Fault Tolerance in Petascale/Exascale Systems: current knowledge, challenges and research opportunities. *Intl Jour of HPC Applications*, Vol. 23, No. 3, Fall 2009.

[7] Schroeder, B. and Gibson, G. Understanding failures in petascale computers. *Journal of Physics: Conference Series*, Vol. 78, (SciDAC 2007).

[8] Pinheiro, E., Weber, W.D., and Barroso, L. A. Failure Trends in a Large Disk Drive Population. In *Proc. of the 5th USENIX Conference on File and Storage Technologies (FAST),* 2007.

[9] Brandt, J., *et al*. Resource Monitoring and Management with OVIS to Enable HPC in Cloud Computing Environments. In *IEEE Intl Sym on Parallel & Distributed Procsng*, 1-8, 2009.

[10] Brandt, J., *et al.* Using Probabilistic Characterization to Reduce Runtime Faults in HPC Systems. In *Proc 8th IEEE Intl Symp on Cluster Computing and the Grid*, 759-764, 2009.

[11] Hsu, C. and Poole, S.W. Power measurement for high performance computing: State of the art. In *Third International Green Computing Conference*, Orlando, FL, July 2011.

[12] Cui, Z., Zhu, Y., Bao, Y., Chen, M. *A Fine-grained Component-level Power Measurement Method*. In *Third International Green Computing Conference*, Orlando, FL, July 2011.

[13] Buennemeyer, T., *et al*. Battery-Sensing Intrusion Protection for Wireless Handheld Computers using a Dynamic Threshold Calculation Algorithm for Attack Detection. In *Proc. of the 40th Hawaii Intl Conference on Systems Sciences*, 2007.

[14] Asante Networks, Inc. *Simple Network Management Protocol*. March 2005. Online: http://www.asante.com/downloads/productdocuments/snmp.pdf