WattsUpDoc: Power Side Channels to Nonintrusively Discover Untargeted Malware on Embedded Medical Devices

Shane S. Clark,[1] Benjamin Ransford,[2] Amir Rahmati,[3] Shane Guineau,[1]
Jacob Sorber,[4] Kevin Fu,[3] and Wenyuan Xu[5]

[1]*University of Massachusetts Amherst,* [2]*University of Washington,* [3]*University of Michigan,*
[4]*Clemson University,* [5]*University of South Carolina, Zhejiang University*

Abstract

Medical devices based on embedded systems are ubiquitous in clinical settings. Increasingly, they connect to networks and run off-the-shelf operating systems vulnerable to malware. But strict validation requirements make it prohibitively difficult or costly to use anti-virus software or automated operating system updates on these systems. Our add-on monitoring system, WattsUpDoc, uses a traditionally undesirable side channel of power consumption to enable run-time malware detection. In our experiments, WattsUpDoc detected previously known malware with at least 94% accuracy and previously unknown malware with at least 85% accuracy on several embedded devices—detection rates similar to those of conventional malware-detection systems on PCs. WattsUpDoc detects malware without requiring hardware or software modifications or network communication.

1 Introduction

Health care networks are composed of general-purpose computers, (e.g., desktop workstations) and embedded devices that perform specific functions and connect to the network for centralized control or configuration. A primary drawback of increasing connectivity is that all devices on the network—including embedded devices—are increasingly exposed to malware [1, 25]. The U.S. Food and Drug Administration has recently acknowledged these risks by issuing a safety communication concerning cybersecurity [27].

Unfortunately, many embedded devices are incompatible with conventional software-based anti-malware mechanisms such as antivirus (AV) programs or networked intrusion-detection systems (NIDS). Traditional embedded devices commonly use custom firmware or OSes for which no antivirus program exists. They are also subject to severe resource constraints that make conventional AV techniques difficult to implement.

There is a second class of embedded medical devices that are built with commodity hardware and software and are thus compatible with AV or NIDS, but their configurations are commonly off limits to their owners because manufacturers will not support third-party software. Some manufacturers explicitly forbid device owners to install OS security updates or antivirus software [2], under the impression that they cannot certify a device's safety if the software configuration changes. This attitude contravenes the FDA's advice in the U.S. [26]. Worse, it precludes the minimally obstructive monitoring that system administrators depend on to detect malware infections.

The fundamental tension for owners of these devices is that they can have the devices they need to perform critical functions, but they cannot adequately protect the devices using conventional, software-based means. Furthermore, the inaccessibility of system information on these systems leaves IT administrators unable to generate meaningful problem reports to submit to public incident-tracking databases—resulting in systematic underreporting of adverse events.

Kramer et al. highlight the disparity between the likely number of adverse security events and the number of available reports [19]. They examined two FDA adverse event databases—populated voluntarily by healthcare facilities and manufacturers—and a Department of Veterans Affairs (VA) database created for internal tracking of security and privacy problems. While the FDA databases yielded almost no direct evidence of security problems, the VA database recorded 207 confirmed malware infections over a 35-month period despite IT administrators' following the advice of manufacturers to keep devices separate; the network implements thousands of ACLs and uses VLANs extensively. This disparity hints that other medical providers may have similar malware problems—but where are the adverse-event reports? Kramer et al. conclude:

Published in Proceedings of USENIX Workshop on Health Information Technologies, 2013

> *While intentional interference may be much less likely to manifest clinically than other types of traditional malfunctions, it is clear that no effective system exists to detect signals of security or privacy problems. This conclusion is confirmed by the sharp contrast of security and privacy problems tabulated by the VA and the security and privacy problems tabulated with FDA databases.*

In light of the above problems, it is desirable for owners to be able to monitor their devices' behavior in meaningful ways to support auditing and adverse-event reports. This paper addresses the challenge of malware on embedded systems by introducing *WattsUpDoc*, a behavior-monitoring system for embedded devices. WattsUpDoc relies on the *side channel* of systemwide power consumption, which leaks information about the system's computing activity without requiring any hardware or software modifications. It is well suited to embedded systems because of their typically constrained state spaces, which manifest as a small set of discrete power-consumption levels and probability distributions.

WattsUpDoc's monitoring is based on machine-learning techniques to match patterns of power consumption. This paper evaluates WattsUpDoc's approach on two similar types of embedded hardware: a pharmaceutical compounder and an industrial-control workstation. Both run variants of the Windows operating system, and both are designed to precisely control important processes. We trained WattsUpDoc on power traces of both normal and abnormal activity, and tested on unlabeled samples of both behaviors. WattsUpDoc flagged abnormal activity on the industrial-control workstation with greater than 98% accuracy, and on the compounder with greater than 94% accuracy. When tested on malware for which WattsUpDoc was *not* trained, the accuracy was 91.3% and 84.2% on the two devices respectively—compared to the 55% detection rate of a commercial antivirus product and the 86% detection rate of a recent behavior-based malware detector [11], yet without requiring software or hardware changes.

Using WattsUpDoc, device owners can gain greater visibility into the behavior of the systems they own. WattsUpDoc can provide preliminary evidence of abnormal behavior, such as malware, signaling the need for further investigation. With better visibility and earlier warnings, WattsUpDoc can help address underreporting of adverse events.

2 Vulnerable Embedded Systems

Embedded systems play a vital role in many operations, from hospitals to industrial facilities where they control physical infrastructure. These systems are commonly "hands off" for customers, i.e., they prohibit customers from changing software configurations, for several reasons. First, manufacturers are unwilling to support software that they did not install and have not validated. Second, customers are disinclined to risk breaking systems that are working, when defending against unclear threats has an unclear benefit. Third, many devices use specialized hardware and software that may not match customers' assumptions.

The key challenge to resolving the tension between functionality and the need to protect increasingly connected devices is finding a way to monitor and validate devices' behavior without modifying them. More specifically, given a device that may contract malware, the challenges are: (1) inferring the device's state without instrumenting or otherwise modifying it; (2) recognizing deviations from these acceptable states, ideally without a database of known deviations; and (3) minimizing the operational burden by avoiding false alarms.

The remainder of this section describes two examples of embedded systems and outlines the threats that malware poses to them.

Medical devices. The category of medical devices encompasses a variety of clinical systems. Over half of these devices include software [10], and many have proven vulnerable to common viruses and malware [24]. Many vulnerable medical devices are directly responsible for patient care, including tissue oximeters, fetal monitors, and pharmaceutical compounders [1].

For this work, we tested a pharmaceutical compounder (Figure 1) we obtained from an auction. A compounder mixes precise amounts of liquid ingredients according to pharmaceutical formulations. The compounder runs Windows XP Embedded and custom mixing software. It includes a network port for loading formulations from the network in multi-compounder environments. It also includes specialized hardware to support its mixing function, including a pump, pressure sensors, and a scale to ensure that mixtures meet weight expectations.

The manufacturer expressly advises against the installation of software updates or other typical protections [2]. According to a security whitepaper by the manufacturer, owners of the compounder should take *some* precautions, namely a dedicated firewall and subnet per device [2].[1] The burden of installing a firewall for each device, and the risk of the firewall interfering with device communication, highlight the need for new approaches to security in this domain. The anecdotal evidence from a VA hospital suggests that this burden raises the bar too high for IT administrators.

[1] The instance of the ExactaMix software on the compounder we tested included network paths (for logfiles, etc.) indicating that the compounder had been connected to a Windows network.

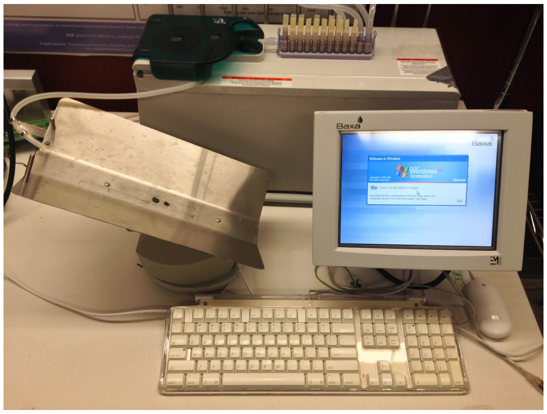


Figure 1: Running on Windows XP Embedded SP2, our Baxa ExactaMix 2400 pharmaceutical compounder is an automated embedded system that mixes liquids to individual specifications for intravenous parenteral nutrition.

Device	**Configuration**
Baxa ExactaMix 2400 compounder	WinXP Embedded, Via 664 MHz , 512 MB RAM
Schweitzer SEL3354 substation computer	WinXP Embedded, Athlon 2600+, 2 GB RAM

Table 1: Devices against which we tested WattsUpDoc.

SCADA systems. In addition to a medical device, we evaluated WattsUpDoc on a Supervisory Control and Data Acquisition (SCADA) device designed for industrial applications; the similar use cases result in similar hardware and software configurations.

SCADA systems comprise hardware and software that monitors and controls industrial processes. This work considers a substation computer, which is a "ruggedized" commodity PC that controls programmable logic controllers (PLCs) and other automation hardware via an array of communication ports. It may also communicate with a larger network via Ethernet or similar. Substation computers typically run "embedded" versions of mainstream operating systems. We tested a substation computer running Windows XP Embedded.

2.1 Threat Model

Because they incorporate both embedded and general-purpose computing devices, medical and SCADA systems are vulnerable to malware targeting generic off-the-shelf systems *and* to more-specific targeted malware akin to Stuxnet [4, 9].

WattsUpDoc does not address targeted threats by determined, well-funded adversaries. Such an adversary with detailed knowledge of the defense mechanisms can design an attack specifically to thwart or evade them. Fortunately, these adversaries appear to be rare; we know of no targeted attacks against medical devices in the wild, and only a few examples of targeted SCADA malware have been publicly acknowledged.

Garden-variety malware, on the other hand, is a clear and present danger to both medical and SCADA systems [19, 25]. We contacted two security professionals at academic medical centers to solicit first-hand perspectives on the types of threats they encounter. Both sources agreed that they have not seen any evidence of targeted attacks against medical devices. One of the two enumerated the top threats in his recent experience, listing three widespread pieces of malware from the past year and the Conficker worm, first identified in 2008. Based on the available evidence, this paper focuses exclusively on flagging, rather than directly stopping, *untargeted* malware threats—those that are not designed specifically to evade power analysis.

We assume an attacker may use software exploits to gain administrator-level access. For devices that are not network-connected, it is important to note that they are potentially exposed to malware if *any* node they interact with can accept outside inputs from, e.g., the Internet or a USB memory stick. We also assume that devices are initially shipped without malware, providing a window in which to train WattsUpDoc.

3 Validating Device Behavior with Power Analysis

Many embedded medical devices share two key properties that make them amenable to nonintrusive monitoring: (1) they perform well-defined, repetitive tasks that should exhibit little variation from run to run; and (2) they draw power from a power outlet. The power outlet can serve as a monitoring point for unmodified hardware.

Many embedded devices perform a small number of repetitive functions, such as actuating an electrical relay, controlling a pump, or collecting sensor readings. Devices based on off-the-shelf OSes (such as our compounder) commonly run a single application that at least conceptually constrains the computer's operation; it is not uncommon for such an application to hide as much of the OS as it can, to give the illusion of a single-purpose computer. As a consequence, the externally visible state space is small.

Components' power consumption as an undesirable side channel is well established [18, 8, 6]. However, side channels can also leak *constructive* information. Many computing devices exhibit systemwide power consumption that scales closely with their workloads. WattsUpDoc uses systemwide power consumption, measured at the outlet, as a proxy for computing activity to nonintrusively detect problems, including malware.

WattsUpDoc design goals. In light of the constraints mentioned above, we set the following explicit design goals for WattsUpDoc:

1. Monitor power *nonintrusively*—do not require device modifications, network connection, or intrude upon their operation.
2. Meet or exceed the malware-detection rate of software-based antivirus products.

To meet the first goal, WattsUpDoc operates exclusively outside the device being monitored, and takes read-only power measurements of which the device is unaware. Section 4 discusses how WattsUpDoc meets the second goal.

A general-purpose computer is complicated in the sense that it has a tremendous number of possible states. We assume that fully inferring the internal state of a computer by inspecting its power consumption is impractical in the general case. However, medical devices are typically dedicated to a single application per machine, effectively constraining the state space to the application's.

Thanks to this simplification of the state space, we can characterize normal behavior for the medical and industrial-control devices we examine in this work.

- A compounder stores chemical recipes (possibly on network shares), mixes chemicals, and verifies the output. It also supports "flushing" to clear its fluid channels of trace chemicals.
- A substation computer periodically collects information from PLCs via wired interfaces, and periodically reports information to higher-level systems.

Activities other than these may appear in traces of the whole system's power consumption, in which case a classifier can flag them as unusual.

While these state descriptions may seem simplistic, the devices we consider have narrow intended purposes. In the case of the compounder, a single custom application opens full-screen at system launch and is intended to be the only user interface to the device. Outside of variations in the mixed chemicals, there are few user interactions that should affect power consumption.

For a concrete example of normal versus abnormal behavior, Figure 2 depicts several power traces of a substation running Windows XP Embedded. At idle, (Figure 2(a)), the system's power consumption is constant (modulo some noise). However, after contracting the widespread "Ramnit" virus that attempts to join a botnet (Figure 2(b)), its power consumption at idle periodically exceeds the normal range. Under the control of an emulated piece of malware, the power consumption at idle exhibits more-pronounced periodic behavior (Figure 2(c)).

3.1 Recognizing Deviations

Antivirus scanners identify malware using signature matching to identify unknown code specimens, requiring an exact (rather than heuristic) match to known malware for a specimen to be flagged as suspicious. The major limitation to this approach is the need to create and distribute signatures for each piece of malware before it becomes widespread. Furthermore, recent work has demonstrated that signature-based antivirus scanners are easily fooled by simple mutations to known malware [15], resulting in false negatives. In the settings we consider, excessive false negatives may have damaging repercussions. For this reason, and also because signature databases require frequent updates from online sources, we seek alternatives to the naïve signature-based approach.

The next section describes a behavior-based approach that offers better detection rates than commercial antivirus software and does not require the device owner to periodically update a blacklist.

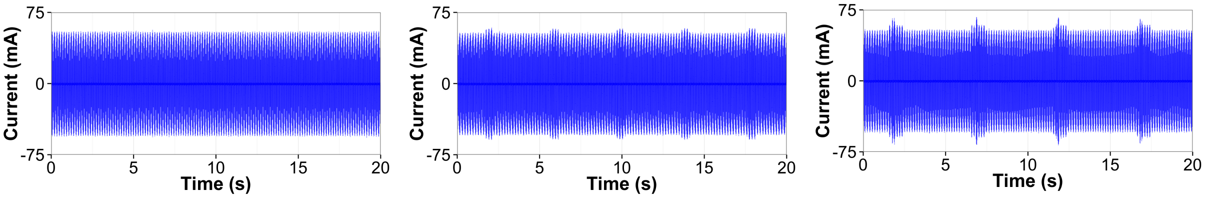
3.2 Classification with Supervised Learning

WattsUpDoc monitors systemwide power consumption at run time. To identify anomalous activity, WattsUpDoc uses a supervised-learning approach that requires traces of normal and abnormal activity. While WattsUpDoc does require some examples of abnormal activity, our experiments in Section 4.2 show that it is not necessary to completely characterize the types of abnormal activity that may occur in the future.

AC power traces are not amenable to straightforward time-series classification. The constant 60 Hz oscillations in the source signal and noise introduced by the power supply make phase alignment a difficult task, thus invalidating many simple distance metrics. For these reasons, WattsUpDoc uses a *supervised-learning* approach to AC power classification that takes both negative (normal) and positive (abnormal) training examples.

Choice of classifiers. Using implementations from the Weka toolkit [13] and libsvm [5], we tested a variety of supervised-learning algorithms for AC trace classification. WattsUpDoc uses the three classifiers that worked best in our experiments; 3-nearest neighbors (3-NN), multilayer perceptron, and random forest.

All three act as binary classifiers that separate normal behavior from abnormal behavior. WattsUpDoc splits each power trace into 5-second chunks to simulate real-time sampling, then applies stratified 10-fold cross val-



(a) Idle

(b) Idle after infection with “Ramnit” malware

(c) Idle after infection with custom malware that captures a screenshot every 5 seconds

Figure 2: AC power traces collected on a substation running Windows XP Embedded in a SCADA testbed.

idation for training and testing. Stratification ensures that each fold in the cross validation process contains approximately the same number of normal and abnormal samples, which is important for repeatable results on biased datasets such as those presented in this paper. In the case of a medical device or other system using general-purpose hardware and software, we focus on malware as the most interesting type of abnormal behavior.

Feature selection and training. After manually examining AC power traces with statistical tools, we selected a set of time-domain features meant to reflect the types of variation that can occur in an AC signal. While performing routine tasks, we expect the distribution of a power trace and the mean power consumed to remain relatively constant over short time periods. Any unexpected software workloads will place extra strain on the hardware, which will in turn consume additional power and affect feature values. WattsUpDoc considers eight time-domain features:

- *mean, variance, skewness, kurtosis:* the 1st, 2nd, 3rd, and 4th moments of the data.
- *Root mean square (RMS):* the square root of the mean squared amplitude;
- The global *minimum* and *maximum* values from all samples in a trace;
- *Interquartile range (IQR):* the difference between the 75th- and 25th-percentile values in the trace; and

WattsUpDoc also uses frequency-domain features. It transforms AC power traces into the frequency domain using the Fourier transform, producing divisions 250 Hz wide. It uses the energy in the ten lowest divisions as features representing periodic content up to 2.5 kHz. Feature f1 represents the energy observed in the interval between 0 and 250 Hz, f2 covers the interval between 250 Hz and 500 Hz, and so on up to f10. Our initial experiments indicated that adding more frequency-domain features did not significantly increase the accuracy of the classifiers. It did, however, increase training time which scales with the number of features.

4 Evaluation

WattsUpDoc’s successful operation depends on its ability to distinguish between normal and abnormal activity after training on power traces of a computer. This section describes our experiments on a pharmaceutical compounder and a SCADA substation computer. Our results can be summarized as follows:

- On each of two general-purpose computers dedicated to a specific set of tasks, WattsUpDoc identifies anomalous activity, including malware, with greater than 94% accuracy (94% on a compounder and 99% on a substation computer).
- WattsUpDoc can be used for soft real-time classification. The supervised-learning approach can operate with a 5-second lag because it processes 5-second chunks and can extract and classify feature vectors in less than 5 seconds at run time.

WattsUpDoc uses binary classifiers that label samples as normal (“negative”) or abnormal (“positive”). The following standard performance metrics apply:

Accuracy, $tp + tn/$ # samples, is the fraction of correctly labeled samples.

Precision, $tp/(tp + fp)$, is the fraction of positively labeled samples whose labels are correct. It measures the classifier’s resistance to false positives.

Recall, $tp/(tp + fn)$, is the fraction of samples that *should* have been positively labeled that *are* correctly positively labeled. It measures the classifier’s resistance to false negatives.

In the above definitions, *tp* and *tn* refer to true positives and true negatives, and *fp* and *fn* refer to false positives and false negatives. A classifier’s precision and recall results provide insight into the *types* of errors the classifier tends to make, rather than counting only the *number* of misclassified samples.

4.1 Experimental Setup

The goals of our equipment design are simplicity, low cost, and modularity (so that devices being monitored need not be modified).

Our previous work on power monitoring [6, 7] described our tracing setup, summarized here. To instrument a standard North American power outlet, we placed a $0.1\,\Omega$ sense resistor (1% tolerance) in series with its neutral terminal. After plugging the device under test into this outlet, we used an Agilent U2356A data acquisition unit (DAQ) to sample the voltage drop across the resistor at a rate of 250 kHz and save the trace to a file. Because we use the DAQ only as a recording device, and the size and cost of the instrumentation is low, we envision a future version of this system that is encapsulated in either an outlet cover or a small power brick with a pass-through outlet.

Compounder. To evaluate our technique on real medical equipment, we purchased a Baxa ExactaMix 2400 compounder via a public auction website. The model we tested runs the manufacturer's compounder software under Windows XP Embedded 2002 Service Pack 2. According to the BIOS boot screen, the compounder has a 664 MHz VIA C5 x86 CPU and 496 MB of RAM.

We gathered AC power traces of the compounder running a wide variety of workloads, including booting, shutting down, flushing, compounding 3 mixtures, infected with our emulated malware, and infected with real-world malware samples. We also gathered traces of the compounder idling, i.e., powered on with the compounder software on screen, but no compounding task running. These traces establish a baseline activity level.

Substation computer. We tested a Schweitzer SEL3354 substation computer [22], both at idle and under a variety of real and simulated malicious workloads. The SEL3354 we tested (circa 2003), was powered via a standard AC outlet, and included an AMD Athlon 64 2600+ processor, 2 GB of RAM, and a Compact Flash card for primary storage in a rack-mountable enclosure. It ran the manufacturer's substation software under Windows XP Embedded. The only hardware difference from an off-the-shelf computer is the addition of 16 RS-232 serial ports intended for use with devices such as PLCs.

We gathered AC power traces of the substation computer that represent most of its typical range of activity, including idling (powered on, Windows desktop on screen, no user interaction once booted), rebooting, infected with our emulated malware, and infected with real-world malware samples.

Malware selection. This paper considers *untargeted* malware in clinical and industrial environments. Untargeted malware finds its way onto off-the-shelf systems via familiar vectors such as infected USB drives and network vulnerabilities.

To test whether our classifiers could detect deviations caused by untargeted malware on the compounder and substation computer, we manually installed twelve of the most-prevalent malware programs of 2011 and 2012 [23, 21], starting with the most prevalent and selecting alternatives when necessary based on sample availability or unsuccessful infection. We could not install antivirus software on the substation computer or compounder because their versions of Windows XP Embedded did not include the necessary framework or installer support. Without a definitive detection mechanism, we considered a malware install "successful" if the malicious executable launched without any user-visible error.

Each trial began with a "clean" snapshot of the original system. While it is difficult to generalize about common malware, those with the greatest prevalence are a reasonable starting point. We also exposed the compounder to the top four most-prevalent malwares identified by one of the healthcare security professionals we consulted. Three out of four failed to infect the compounder's "embedded" variant of Windows, presumably because of assumptions encoded in the malware, but one infection occurred. We chose not to alter any software on the compounder to make it vulnerable to specific threats, to avoid inadvertently changing other properties.

To test detection of common malware archetypes, we also wrote emulated malware designed to mimic routine misbehavior. Specifically, we implemented a keylogger, a pop-up dialog launcher that opens dialog boxes (to emulate adware), and a screen grabber that saves screenshots at a fixed interval (to emulate common exfiltration techniques).

4.2 Experimental Results

WattsUpDoc identifies known malware—malware it was trained to recognize—with 94% accuracy on the compounder and 99% accuracy on the substation computer. Additionally, WattsUpDoc identifies unknown malware—unlabeled samples of a malware infection that were not in the training set—with 88.5% accuracy on the compounder and 84.9% accuracy on the substation computer, which is comparable to the 86% detection rate of a state-of-the-art behavior-based malware detector [11].

4.2.1 Pharmaceutical Compounder

We gathered traces of the ExactaMix compounder while performing a variety of real-world tasks, booting and shutting down, and while running a wide variety of emulated and authentic malware workloads. We were able to profile the compounder while it performed special-

Classifier	Accuracy	Precision	Recall
Compounder			
3-NN	93.6%	89%	78%
Perceptron	94.4%	99%	73%
Random Forests	94.2%	93%	77%
Substation			
3-NN	99.5%	100%	100%
Perceptron	99.5%	100%	100%
Random Forests	99.5%	100%	100%

Table 2: WattsUpDoc distinguishes among our test workloads: accuracy, precision, and recall for our AC devices when we trained WattsUpDoc on samples of *all* of our test workloads.

ized actions including mixing ingredients according to programmed recipes and flushing the fluid inlets. The classification task was to separate normal behavior (defined as idle, booting, shutdown, or compounding tasks) from the malware workloads. We used stratified 10-fold cross validation, which enforces approximately the same positive/negative composition in the training and testing sets to avoid biases. There were 2343 total samples tested, 1845 positive (infected) samples and 497 negative samples. We used many more positive samples than negative samples because gathering negative samples required manual interaction with the compounder and frequent solution refills. In a clinical deployment, regular use would provide a ready source of negative samples. All three of the classifiers achieved at least 93% accuracy. Table 2 summarizes the full results. This experiment demonstrates that WattsUpDoc's classifiers can reliably separate different workloads given training samples of *all* workloads expected at testing time.

We re-trained WattsUpDoc on half of the compounder dataset after removing all samples of half of the real malware variants. We then tested WattsUpDoc using only normal samples and those drawn from the malware variants *not* used at training time. That is, the malware variants used at training and testing time were randomly assigned disjoint sets. We partitioned, trained, and tested ten times to avoid a single biased experiment. Finally, we implemented majority voting among WattsUpDoc's classifiers to produce a single label for each sample. This experiment presents a more realistic deployment scenario, one in which WattsUpDoc has access to some malware samples at training time but has not been trained on all of the malicious workloads seen at testing time. Table 3 summarizes the results, which show that the accuracy decreases by approximately 5%, but is still comparable to the state-of-the-art in behavior-based malware detection. The precision is greater than 93%, meaning that WattsUpDoc produced few false positives, which could lead to unnecessary downtime in a clinical setting.

Device	Accuracy	Precision	Recall
Compounder	88.5%	93.5%	92.1%
Substation comp.	84.9%	98.3%	80.8%

Table 3: Mean accuracy, precision, and recall over 10 runs for each dataset with malware samples randomly partitioned into two disjoint sets. One set was used for training and the other for testing.

Feature ranking. To evaluate which features best separate samples from each class, we used Weka's information-gain attribute evaluator to rank the features by their contribution to classification (measured as mutual information between each feature and the class label). Mutual information is the difference between the entropy of the feature and the entropy of the feature conditioned on the class label, expressed as: $I(feature;class) = H(feature) - H(feature|class)$ where H is the entropy and I the mutual information. This ranking is feature set dependent, but not classifier dependent.

For this dataset, we found that the top five features in order were mean, RMS, skewness, variance, and maximum. None of the frequency-domain features appear in the list of top features. Rather, all of the most valuable features for this dataset pertain to the average power draw and slow-moving changes in that power draw. The dynamic range of the mixing and flushing traces suggest that the compounder's pump is responsible for the relative utility of these features. When the pump switches on, the compounder's total power consumption increases by more than 30%.

4.2.2 Substation Computer

We gathered samples of the SEL3354 substation computer's power consumption while running the Schweitzer software but otherwise idling, rebooting, and running a wide variety of emulated and authentic malware workloads. The classification task was to separate normal behavior (defined as idle or reboot traces) from abnormal behavior (the malware workloads). As for the compounder dataset, we used stratified 10-fold cross validation for training and testing. There were 2634 total samples tested, 364 negative samples and 2270 positive (infected) samples. All three of the classifiers performed nearly perfectly, with 99.5% accuracy and precision and recall both rounding to 100%.

As for the compounder, we re-trained WattsUpDoc on

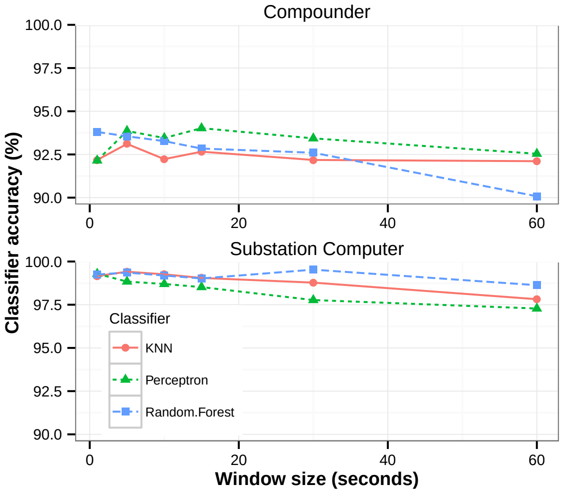


Figure 3: The effect on accuracy as the size of the window over which features are calculated increases. 5-second windows produce the highest accuracy for all three datasets.

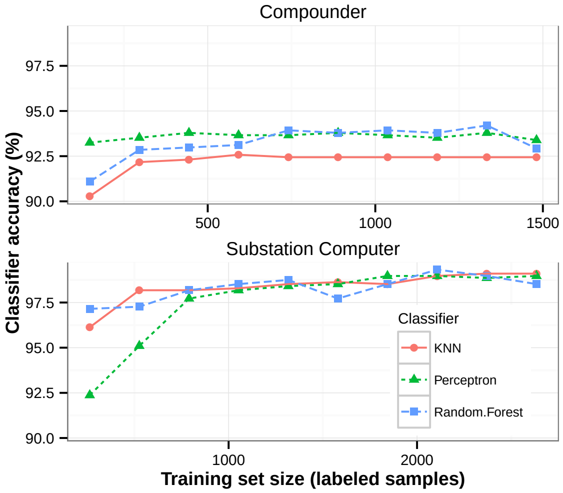


Figure 4: The effect on accuracy as the number of training samples increases. The point of diminishing returns appears to be approximately 500 for the compounder dataset and 1000 for the substation dataset.

the substation computer dataset ten times after randomly partitioning the malware variants into disjoint training and testing sets to simulate a more realistic deployment scenario. Table 3 summarizes the results. Once again, the accuracy decreased versus the accuracy on known malware samples, by approximately 15% for this dataset. The recall decreased to 80.8%, meaning that nearly 20% of the malware samples went undetected, but the precision remained high (98.3%), which indicates that samples WattsUpDoc flagged as infected actually represent a malware infection with high probability. A possible explanation for the performance difference with unknown malware on the substation dataset is that we gathered traces for fewer malware variants using the substation computer (8 versus 12), so fewer training samples were available for this experiment.

Feature ranking. For this dataset, we found that the top five features in order were f3 (total energy between 500 and 750 Hz), maximum, f4 (total energy between 750 and 1000 Hz), RMS, and mean. The highest-ranked features all appear to be related to the overall power consumption or frequency components. The frequency components are indicative of rapid changes in power consumption, which cause power supplies to induce different noise patterns on the circuit. The substation computer has no special-purpose hardware that exhibits slow and obvious state changes, unlike the compounder. This hardware disparity provides a plausible explanation for the relative value of the frequency-domain features in the substation computer dataset.

4.3 Maximizing Classification Accuracy

Two parameters directly affect WattsUpDoc's classification accuracy: the *window size* over which features are computed, and the *training set size*.

Window size. To systematically select the size of the window over which WattsUpDoc calculates features, we tested six values ranging from 1 second to 60 seconds. We tested each window size once using the full dataset for each device. Excessively narrow windows could fail to accurately capture longer-running operations on devices like the compounder and excessively wide windows could lower the effective signal strength of short-lived operations, allowing malware to go undetected. For both datasets, we found that 5-second windows produced near-maximal accuracy for all three classifiers and that the accuracy tended to decrease slowly as the window size increased. Figure 3 summarizes the results.

Training set size. To determine how many training samples WattsUpDoc requires to reliably separate normal from abnormal behavior, we randomly selected one third of all samples from each dataset for testing and trained on 10–100% of the remaining samples in 10% increments. Each training set size was tested once per device and the training sets contained samples of both normal and abnormal operation. Figure 4 summarizes the results, which show diminishing returns in accuracy for the compounder after approximately 500 samples and for the substation computer after 1000 samples.

5 Related Work

Cárdenas et al. provide an overview of the challenges endemic to SCADA systems, including the difficulty of modifying them once they are deployed, and propose an approach to SCADA-device anomaly detection that uses knowledge of the system being monitored to develop a template of normal behavior [3]. Unlike our work, Cárdenas et al. evaluate their system in simulation and rely on directly observing the physical devices attached to a control system as inputs and outputs.

Similar to our approach, Cheung et al. advocate model-based malware-detection techniques for Modbus TCP networks and develop an IDS [28]. Beside our emphasis on power traces instead of network traces, we do not make any assumptions about the protocols in use in the systems that we seek to protect.

Khan et al. use power tracing to diagnose problems with sensor nodes deployed in the field [16]. Our work focuses on a wider variety of devices and must consider adversarial scenarios in which illegitimate patterns may hide in legitimate ones.

AC power event recognition. Previous work has explored *coarse-grained* energy monitoring that detects electrical activity (such as appliances' on–off transitions) at a household level [14, 12]. WattsUpDoc takes a finer-grained view of a single complex device with an emphasis on anomaly detection rather than energy reporting.

Our work is conceptually related to previous work analyzing power traces from embedded systems. In particular, Enev et al. refined the ElectriSense concept by studying the correlation of EMI with video signals being played on modern high-definition televisions. HDTVs are comparatively simple embedded systems and are amenable to simple models, whereas many medical devices are not.

Power analysis and malware. Kim et al. and Liu et al. propose and evaluate power analysis as a malware detection mechanism for mobile phones [20, 17]. Liu et al. present VirusMeter, which uses explicit modeling of a small number of user actions to build a state machine and flag later anomalous power consumption using a variety of classifiers. Kim et al. apply a similar approach that requires offline measurements of the device under test to create a suitable model for later use. Kim et al. build signatures for individual pieces of malware. Like VirusMeter, their system relies on power consumption information provided by the phone that they seek to protect. Our techniques do not require modifications to the device under test and do not use potentially untrustworthy APIs to obtain power consumption information.

This work conceptually extends our recent explorations of AC power analysis that classified power signatures of webpages with a frequency-domain classifier and proposed applying power analysis to the problem of malware detection [7, 6]. The webpage analysis focused on general-purpose commodity computing devices [7]; we extend this approach, applying different classification techniques and features to a new application on embedded devices that exhibit smaller state spaces.

6 Discussion and Extensions

While the power-analysis techniques in this paper show some promise for revealing malicious activity on embedded or hard-to-change devices, they are not a complete solution to the problem of malware on these devices. This section discusses some of the issues still surrounding real-world deployment of WattsUpDoc and the generalizability of our technique.

Deployment Scenarios. Since our measurement mechanisms are low cost and nonintrusive, we envision pairing embedded devices with measurement points. As mentioned in Section 4.1, an AC measurement point could be a simple "wall wart" form-factor brick with one or more pass-through outlets into which devices could be plugged, or even an in-wall outlet box that looks like a standard outlet plate. In a networked environment, measurement points would stream their readings to a centralized computer for classification and logging. To save traffic, each measurement point could use a low-cost microcontroller to perform feature extraction, then send only sets of feature vectors. Instrumenting only select devices, e.g., those that owners know run outdated software or require network connections, is another possibility that would reduce the management burden while providing increased visibility.

The best strategy for balancing false positives and false negatives in a clinical or industrial setting is not yet clear. It is a waste of resources for device maintainers to follow up on false positives, but decreasing system sensitivity too far runs the risk of ignoring valid alarms. By designing WattsUpDoc primarily as an offline reporting tool intended to gather evidence of suspicious activity, we hope to minimize the risk of introducing complexity without providing valuable new evidence of infections.

NIDS. Networked intrusion detection systems (NIDS) are an important monitoring tool for many sites. Unfortunately, most NIDS' detection strategies depend on a certain minimum level of visibility into the systems being monitored, such as access to log entries or the filesystem. As described in the introduction, some vendors prohibit any modifications at all to their systems—leaving NIDS unable to monitor their behavior. We intend WattsUpDoc to be complementary with NIDS; in fact, it could feed its detection results into NIDS monitoring to provide visibility into systems that otherwise would be unmonitored.

Generalizability. The systems we evaluated run "embedded" versions of the Windows operating system. The key difference versus consumer-grade workstations is that the medical and SCADA systems are designed for specific applications that constrain the computer's state space—and consequently also constrain its power consumption. In contrast, off-the-shelf PCs lack software or policy restrictions to prevent them from running many tasks simultaneously or executing new code. Without a consistent base set of known-good behaviors on a PC, WattsUpDoc would likely raise false alarms because of an inconsistent or inaccurate internal model.

Although it may not generalize to commodity PCs, WattsUpDoc would likely work well on other embedded devices that rely on firmwares instead of true operating systems. With little or no concurrency, and with state spaces that resemble finite state machines, many embedded devices are comparatively simple to characterize.

7 Conclusion

Safety-critical medical devices need better protection against conventional malware. Protecting these systems without modifying their strictly validated embedded software is a key challenge. WattsUpDoc is the first behavior-monitoring system for embedded devices that monitors *systemwide* power consumption. Designed to detect aberrant behavior on devices that exhibit constrained state spaces, WattsUpDoc uses machine learning to model permissible behavior and detect deviations. WattsUpDoc requires no hardware or software modifications to the devices it monitors. We tested WattsUpDoc on a pharmaceutical compounder and a similar industrial-control (SCADA) system. When we infect these systems with common malware, WattsUpDoc flags the anomalous behavior with accuracy of 94% and 99.5% on the compounder and substation computer respectively. Against malware it was not trained to recognize, WattsUpDoc's accuracy degrades to 88.5% and 84.9% for the compounder and substation computer respectively—still comparable to state-of-the-art behavior-based malware detection. The intuition behind the high accuracy, precision, and recall is that embedded medical systems are designed to do one thing well and repeatedly—unlike general-purpose computers. Malware causes subtle changes to power consumption that machine learning techniques can then identify.

Acknowledgments

We thank the members of the SPQR group for feedback on drafts. We thank Tim Yardley, the Information Trust Institute at UIUC, Phil Dunbar, and Industrial Defender for allowing us access to their SCADA hardware. We thank Katarzyna Olejnik for assistance in analyzing traces. This material is based on work funded by the National Science Foundation under Grant No. CNS-1331652, GEO1124657, CDS0845671, and S121000000211. This publication was made possible by Cooperative Agreement No. 90TR0003/01 from the Department of Health and Human Services. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or HHS.

References

[1] MAUDE Adverse Event Report. `http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfmaude/Detail.CFM?MDRFOI__ID=1621627`, Loaded Nov. 2012.

[2] BAXA CORPORATION. Preventing cyber attacks. `https://btsp.baxa.com/Sales%20Portal/ExactaMix/Preventing%20Cyber%20Attacks.pdf`, Loaded Oct. 2012.

[3] CÁRDENAS, A. A., AMIN, S., LIN, Z.-S., HUANG, Y.-L., HUANG, C.-Y., AND SASTRY, S. Attacks against process control systems: risk assessment, detection, and response. In *ASIACCS* (Mar. 2011).

[4] CÁRDENAS, A. A., AMIN, S., AND SASTRY, S. Research challenges for the security of control systems. In *HotSec* (July 2008).

[5] CHANG, C.-C., AND LIN, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology 2*, 3 (Apr. 2011).

[6] CLARK, S. S., RANSFORD, B., AND FU, K. Potentia est scientia: Security and privacy implications of energy-proportional computing. In *HotSec* (Aug. 2012).

[7] CLARK, S. S., RANSFORD, B., SORBER, J., XU, W., LEARNED-MILLER, E., AND FU, K. Current Events: Identifying Webpages by Tapping the Electrical Outlet. Tech. Rep. UM-CS-2011-030, Dept. of Computer Science, UMass Amherst, July 2012.

[8] ENEV, M., GUPTA, S., KOHNO, T., AND PATEL, S. Televisions, video privacy, and powerline electromagnetic interference. In *ACM Conference on Computer and Communications Security* (Oct. 2011).

[9] FALLIERE, N., MURCHU, L. O., AND CHIEN, E. W32.Stuxnet dossier. `http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf`, Feb. 2011.

[10] FARIS, T. H. *Safe and Sound Software: Creating an Efficient and Effective Quality System for Software Medical Device Organizations.* ASQ Quality Press, Mar. 2006.

[11] FREDRIKSON, M., JHA, S., CHRISTODORESCU, M., SAILER, R., AND YAN, X. Synthesizing near-optimal malware specifications from suspicious behaviors. In *IEEE Symposium on Security & Privacy* (May 2010).

[12] GUPTA, S., REYNOLDS, M. S., AND PATEL, S. N. ElectriSense: Single-point sensing using EMI for electrical event detection and classification in the home. In *UbiComp* (Sept. 2010).

[13] HALL, M., FRANK, E., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., AND WITTEN, I. H. The WEKA data mining software: An update. *SIGKDD Explorations 11*, 1 (2009).

A Malware Samples Used to Test WattsUpDoc

Malware	Activity	Devices
almanahe	Download and execute files	Compounder
autorun	Open port and IE instances	Compounder
bamital	Disable system programs	Substation
bredolab	Download other malware, crash system, unhook API	Compounder
delf	Allow remote access	Compounder
dorkbot	Download other malware, Initiate DOS attacks, Steal personal info	Compounder
fakeav	Download other malware, Ransomware	Compounder
kolab	Spam IRC channels	Both
mabezat	Infect host files	Substation
ramnit	Run malicious routines	Substation
sality	Infect and delete host files	Both
sillyfdc	Disable services	Both
virut	Infect host files, create backdoor	Both
zbot	Steal personal info, create bot	Both
cryptic	Download other malware	Compounder

Table 4: Windows malware representing unusual device behavior.

[14] HART, G. W. Nonintrusive appliance load monitoring. *Proceedings of the IEEE 80*, 12 (Dec. 1992).

[15] JANA, S., AND SHMATIKOV, V. Abusing file processing in malware detectors for fun and profit. In *IEEE Symposium on Security & Privacy* (May 2012).

[16] KHAN, M. M. H., ET AL. Diagnostic powertracing for sensor node failure analysis. In *IPSN* (Apr. 2010).

[17] KIM, H., SMITH, J., AND SHIN, K. G. Detecting energy-greedy anomalies and mobile malware variants. In *MobiSys* (June 2008).

[18] KOCHER, P., JAFFE, J., AND JUN, B. Differential power analysis. In *CRYPTO* (Aug. 1999).

[19] KRAMER, D. B., BAKER, M., RANSFORD, B., MOLINA-MARKHAM, A., STEWART, Q., FU, K., AND REYNOLDS, M. R. Security and privacy qualities of medical devices: An analysis of FDA postmarket surveillance. *PLoS ONE 7*, 7 (July 2012), e40200.

[20] LIU, L., YAN, G., ZHANG, X., AND CHEN, S. Virusmeter: Preventing your cellphone from spies. In *RAID* (Sept. 2009).

[21] RAINS, T. Operating system infection rates: The most common malware families on each platform. https://blogs.technet.com/b/security/archive/2013/01/07/operating-system-infection-rates-the-most-common-malware-families-on-each-platform.aspx, Jan. 2013.

[22] SCHWEITZER ENGINEERING LABORATORIES, I. *SEL-3354 Embedded Automation Computing Platform: Instruction Manual*, Jan. 2011.

[23] SYMANTEC CORPORATION. Malicious code trends. https://www.symantec.com/threatreport/topic.jsp?id=malicious_code_trends&aid=top_malicious_code_families, Loaded July 2012.

[24] TALBOT, D. Computer viruses are "rampant" on medical devices in hospitals. http://www.technologyreview.com/news/429616/computer-viruses-are-rampant-on-medical-devices-in-hospitals/, Oct. 2012.

[25] U.S. DEPARTMENT OF HOMELAND SECURITY. ICS-ALERT-12-046-01A—Increasing threat to industrial control systems, Oct. 2012.

[26] U.S. FOOD AND DRUG ADMINISTRATION. Reminder from FDA: Cybersecurity for networked medical devices is a shared responsibility. http://www.fda.gov/MedicalDevices/Safety/AlertsandNotices/ucm189111.htm, Nov. 2009.

[27] U.S. FOOD AND DRUG ADMINISTRATION. Cybersecurity for medical devices and hospital networks: FDA safety communication. http://www.fda.gov/Safety/MedWatch/SafetyInformation/SafetyAlertsforHumanMedicalProducts/ucm357090.htm, June 2013.

[28] VALDES, A., AND CHEUNG, S. Communication pattern anomaly detection in process control systems. In *IEEE Conference on Technologies for Homeland Security* (May 2009).