Intrusion Monitoring in Process Control Systems

Alfonso Valdes
Computer Science Laboratory
SRI International
Menlo Park, CA 94025
alfonso.valdes@sri.com

Steven Cheung
Computer Science Laboratory
SRI International
Menlo Park, CA 94025
steven.cheung@sri.com

Abstract

To protect process control networks from cyber intrusions, preventive security measures such as perimeter defenses (for example, network firewalls and demilitarized zones) and secure versions of process control network protocols have been increasingly adopted or proposed. Although system hardening and fixing known vulnerabilities of existing systems are crucial to secure process control systems, intrusion monitoring is essential to ensure that the preventive measures are not compromised or bypassed. Our approach involves a multilayer security architecture for monitoring process control systems to achieve accurate and effective situational awareness. Also, we leverage some of the characteristics of process control systems such as the regularity of network traffic patterns to perform intrusion detection, with the potential to detect unknown attacks. To facilitate human analysts to gain a better understanding of anomalous network traffic patterns, we present a visualization tool that supports multiple user-customizable views and animation for analyzing network packet traces.

1. Introduction

Modern infrastructure systems in manufacturing, transportation, and the energy sector depend on digital control systems for safe and efficient operation. Examples of digital process control systems include Distributed Control Systems (DCSs) and Supervisory Control and Data Acquisition (SCADA) systems used in electric power generation and distribution, oil and gas (O&G) refining, and pipelines. Early digital automation systems were isolated and used purpose-built protocols, networks, and operator interfaces. For reasons of cost effectiveness, modern systems increasingly use commercial off-the-shelf (COTS) network technologies such as TCP/IP. Increasingly sophisticated devices (such as remote terminal units (RTUs), and programmable logic controllers (PLCs)) now come with an embedded operating system and Ethernet port and a web interface for configuration. The operator workstation (human-machine interface, or HMI) is now typically a Windows platform. Moreover, for reasons of timely business access to control system parameters, control systems are increasingly connected to corporate systems, preferably through a demilitarized zone (DMZ) that prevents direct access to the control system from the corporate network. Control systems are thus in some sense coming to resemble enterprise IT computer systems, and are increasingly connected to those systems.

1.1. Motivation

The migration of Process Control Systems (PCSs) to use COTS technologies and the connection to enterprise systems have led to great gains in economic efficiency. There is some concern, however, that these trends have exposed control systems to cyber attack. The consequences of a successful cyber attack on a PCS are potentially much more serious than those of attacks on corporate systems, depending on the nature of the process under control. A cyber compromise of a PCS can lead to destruction of expensive equipment [2], which can lead to loss of production that can last far longer than the time required to rebuild the compromised computer assets. Such an attack can also result in release of hazardous materials into the environment, and in extreme cases loss of life.

The need for cyber security in PCSs is thus arguably greater than the same need in enterprise IT systems. Also, control systems pose special challenges. For example, unlike the priorities of the security objectives for a typical enterprise system, PCS operators are most concerned about maintaining availability and integrity as these most directly impact safe and efficient operation of the underlying process, and typically less concern about confidentiality. In addition, PCSs often comprise a mix of generations of equipment, and older equipment may not be able to support modern security measures. PCS HMI platforms tend to lag enterprise systems as far as OS version and patch level. With respect to adoption of enterprise best practices in security, control system cyber assets tend to lag behind enterprise systems. Given these difficulties, it is especially important to define and secure Electronic Security Perimeters (ESP) in control systems. Firewalls, switched networks, and DMZs are increasingly deployed, in accordance with, for example, the National Electric Reliability Corporation Critical Infrastructure Protection Standards (NERC CIPS [9]). But this is only part of the picture; we consider PCS monitoring an essential complementary defense to ensure that perimeter defenses have not been breached or bypassed. Also, monitoring can be useful for detecting probes, failed attacks, and misuse by insiders.

1.2. Approach

There are several key ingredients in our approach. First, we employ a multilayer monitoring and correlation architecture (depicted in Figure 1) to achieve situational awareness, which involves monitoring system events at multiple levels—device, network, and host levels—to enable more accurate and effective detection of attacks. Also, event correlation is performed at multiple levels—control center, utility, and sector levels—to achieve scalable situational awareness at different abstraction levels. Event correlation is not only used to achieve event/alert reduction, but also used to improve detection accuracy and coverage. For example, anomalous events that initially appear to be process (as opposed to security) related may be more suspicious if they correlate with certain alerts generated by intrusion detection sensors.

Model-based detection is a key detection approach we use. In model-based detection, we develop models that characterize the expected behavior of system components, and detect attacks that cause violations of these models. Denning's anomaly detection [3], Ko et al's specification-based intrusion detection [6], and Forrest et al's work [5] on using system call sequences to detect intrusions are notable examples of this approach. Unlike a more commonly used approach, based on matching attack signatures, knowledge about the attacks is less crucial in the model-based approach. Despite its potential of detecting unknown attacks, model-based detection is not widely used in enterprise networks, because of the difficulties and the costs of creating accurate models. We observe that process control systems tend to have more predictable network communication patterns, have relatively static network and host configurations, and typically use simpler protocols. Thus model-based detection may be applicable for monitoring process control systems.

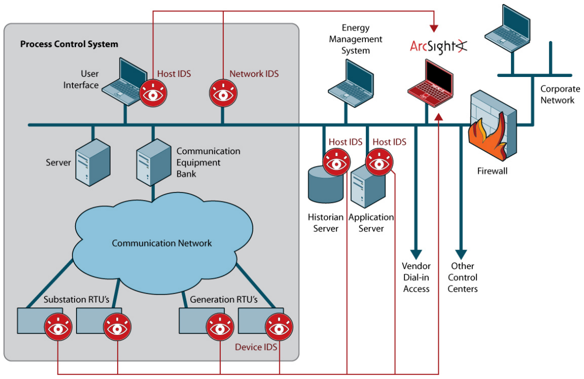


Figure 1. Control Center Level View of the Detection and Event Correlation Framework

For analyzing anomalous network traffic patterns detected in process control networks, we are exploring the use of visualization to facilitate human analysts to gain a better understanding of the anomalies. To this end, we have been developing a tool, called WholeNet viewer [11], that provides multiple user-customizable views and animation to visualize network traffic data. WholeNetViewer provides a compact "data cube" metadata archive that can be used to quickly summarize a data trace of a communication pattern for a time period. Beyond visualization, we will apply pattern anomaly detection to the results of WholeNetViewer, which is discussed in more detail in Section 3.

2. Architecture

This section presents a layered monitoring and event correlation architecture for process control systems. Figure 1 depicts the architecture for an exemplary control center. The architecture consists of several types of components. At the lowest layer, we have intrusion detection sensors monitoring at the device, network, and HMI host levels. These sensors report to event correlation engines, based on ArcSight's Security Information Event Management (SIEM) framework (`www.arcsight.com`) at control center level. At the sector-wide level (not shown in Figure 1), privacy-preserving security incident sharing and correlation are performed across control centers to detect security incidents that impact multiple sites in the sector.

2.1. Intrusion Detection

Intrusion detection sensors monitor the activities on the process control systems, and provide timely and accurate alerting in the case of attempted cyber attacks against the control systems. Attackers on the Internet may access enterprise networks of infrastructure asset owners, using the enterprise as a steppingstone to attack the asset owner's control networks. Attackers may also exploit poorly configured access pathways (for example, vendor access mechanisms using known access credentials), or attack sophisticated field devices such as RTUs and PLCs, increasingly featuring embedded operating systems and Internet protocol connectivity.

We use a model-based detection approach, complemented by conventional signature-based detection. In model based detection, we develop models to characterize the acceptable/expected behavior of the application processes, machines, and users in the system, and monitor the behavior of these components to detect attacks that cause violations of the models.

We use a broad definition of model-based detection, which includes specification-based detection, change detection, and statistical anomaly detection. These various techniques differ in the degree of automation (e.g., manually specifying the models based on protocol specification versus machine learning by observing system behavior[1]), the abstraction levels of the system at which the models specify (e.g., for network traffic, one may construct models based on IP header fields or on higher-level protocol fields), the languages used to specify them, and the techniques used to compare observed behavior with the models to detect violations (e.g., deterministic versus probabilistic approaches).

Compared with signature-based detection—a commonly used intrusion detection approach, which involves developing attack signatures that capture the key characteristics of (known) attacks and detecting system behavior that matches those signatures—model-based detection offers the potential for detecting unknown attacks.

However, model-based detection is not widely used in enterprise systems, because it is generally difficult to develop models that accurately capture the expected behavior, and can be efficiently used to perform intrusion detection. We argue that the model-based approach is more feasible for process control systems, because these systems tend to have a small and static set of applications, regular and predictable communication patterns, and simpler protocols.

To date, we have investigated several detection techniques and performed an experimental validation [1]. Specifically, we have developed protocol-level models for two protocols widely used in control networks, namely, Modbus TCP[2] [7, 8], and DNP3 over TCP/IP[3] (`www.dnp.org`).

These models specify the expected values of packet fields and their relationships. For example, one of our models specifies the allowable Modbus function codes (corresponding to Modbus services). Modbus requests that contain an "unexpected" function code—e.g., attempting to exploit a backdoor or an obscure, insecure service, or performing a denial-of-service attack—can be detected using the model.

Based on these models, we have developed rulesets for Snort [10], a popular network intrusion detection system (IDS), for detecting packets that violate them. Note that Snort is typically used for signature-based detection, matching network traffic against a set of attack signatures. Our work uses Snort in a different way for performing model-based detection—After developing models to characterize the expected behavior, we then develop Snort rules to detect the "complement" of the models. We have also formally specified the characteristics of Modbus devices using the PVS specification language [4]. A future work item is to investigate employing executable specifications for intrusion detection, bypassing the error-prone and time-consuming manual process of converting the models into IDS rulesets.

Another detection technique employs a heuristic approach for learning the models pertaining to the availability of servers and services for Modbus TCP. The basic idea is that when an interesting event (e.g., new Modbus unit ID detected or a change in the status of a Modbus service) is observed for the first time, a report is generated for the state transition. To this end, we have implemented two intrusion detection sensors, namely, EMERALD Bayes sensor [12] and EModbus. The former contains a TCP-level service discovery component, which learns active services on a monitored network, and as these are discovered it maintains a Bayes instance that rapidly detects when the service is down. EModbus discovers supported function codes on the Modbus servers. Service discovery and monitoring of service state are directly useful from a security standpoint and

[1] In addition to characterizing correct and expected behavior a priori, pattern anomaly detection can be used to learn discrete communication patterns and alert to novel or unusual observations.

[2] A Modbus client (also called master) may send a request message to a Modbus server (also called slave). A Modbus request contains a function code, corresponding to the service requested (e.g., read a 16-bit register, or perform diagnostics for the server), and may include a list of arguments (e.g., addresses of data items). After the server receives and processes the request, it sends a response message back to the client.

[3] DNP3 (Distributed Network Protocol) is commonly used in electric utilities to enable data acquisition and control for RTUs and PLCs.

can indirectly help explain other events. A new service or function code in a system that has been in stable operation for some time is suspicious. A Modbus device should not suddenly start responding to a previously unseen function code. Monitoring service state gives a different view of system health from that obtained directly from the control system itself. This can confirm adverse conditions and motivate prompt remedial action, even if the adverse state is not due to malice. Alerts in this case are threaded so that the monitoring system does not generate multiple alerts for the same condition, but rather periodically updates its status until the condition is remedied.

Also, we have developed models to specify the network traffic patterns of an examplary process control network testbed [1], developed by the Sandia National Laboratories (SNL), as part of the PCS security project of the Institute for Information Infrastructure Protection (`http://thei3p.org`). The testbed is equipped with Modbus devices running on a process control network, and contains a demilitarized zone and network firewalls separating a corporate network from the process control network. Based on the models, we developed Snort rules for detecting violations of the traffic patterns for the testbed. This is motivated by the observation that the network traffic patterns in process control systems are typically regular and predictable. Based on the applications running on the machines in a process control network, one can characterize the expected communication patterns among them. For example, in the network testbed, the Modbus server may communicate only with a specified set of Modbus clients. If a Modbus server is compromised and attempts to attack other hosts, the network IDS may detect the anomalous communication pattern and generate an alert for the violation.

These model-based intrusion detection sensors have been incorporated in a multialgorithm network intrusion detection and alert correlation appliance, which has the following features:

- Multialgorithm, multilevel analysis
- Stateful packet reassembly and protocol analysis engine
- Bayesian protocol anomaly detection engine [12]
- Snort, with a ruleset configured to complement the other components of the EMERALD sensor suite enhanced with the PCS ruleset from Digital Bond (`www.digitalbond.com`)
- Probabilistic alert aggregation engine [13]

The appliance uses multiple algorithms to adaptively determine the control system behavior, detect, and report significant deviations. The appliance also accepts and correlates alerts from other sensors in the PCS network (including itself), which can then be viewed via an Alert Management Interface (AMI). The appliance monitors one or more network segments by sniffing traffic over a passive network interface (so it is itself invisible to network probes and cannot engage in a TCP connection) connected to the span port of a router or switch. Reporting and configuration take place through a second interface, ideally connected to a network dedicated to security functions.

We have performed an experimental validation for the multialgorithm intrusion detection approach [1]. In this experiment, SNL developed a multistep attack scenario for the testbed. In the attack scenario, an adversary first compromises a machine in the corporate network, typically connected to the Internet. The adversary then takes control of the historian machine in the demilitarized zone, using which she launches a successful attack against the historian machine in the process control network. From that machine, the adversary performs reconnaissance and attacks the Modbus servers and other hosts in the process control network. In the experiment, different detection algorithms—signature-based, model-based, and Bayes—detected different aspects of the multistep attack scenario, and none of them can be removed without affecting detection coverage.

2.2. Security Information Event Management

A common problem of deploying extensive intrusion detection in infrastructure systems, as in enterprise systems, is the large number of IDS alerts, many of which are false positive or indicative of low-level and failed threats. Alert correlation combines related alerts into security incidents, prioritizes incidents with respect to their likely impact on the system mission, and presents a coherent view to a security analyst to enable timely and effective countermeasures.

In partnership with ArcSight, we plan to design and implement an event correlation framework to provide a novel SIEM capability for infrastructure systems. By enabling ArcSight to comprehend control system events together with IDS alerts, we envision that we can achieve better detection coverage and accuracy than using IDS alerts alone.

Digital control systems monitor and control an underlying physical process such as flow or current. As such, they provide console displays of process-related alarms, such as pressure or flow rate out of some specified range. By developing correlation models for IDS alerts, control system/process alarms (which may not be considered security related per se, but appear so when correlated with an intrusion or attack), and possibly alarms from physical perimeter protection systems, our SIEM framework will explore the degree to which these process alarms correlate with IDS alerts to provide a content-rich security situational awareness picture.

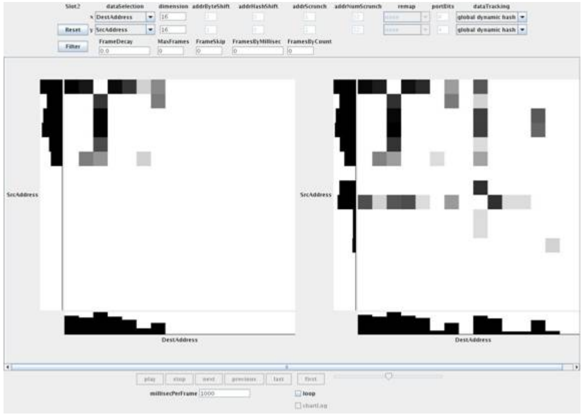
In our architecture, we envision multiple installations of the SIEM capability, with these SIEM components organized hierarchically at multiple levels—control center, utility (multiple control centers, considering threats at the enterprise/control system interface), and among utilities.

3. Visualizing Communication Pattern Anomalies

Based on an observation that the network traffic patterns in process control systems are typically regular and predictable, we are exploring techniques to detect and visualize intrusions that exhibit anomalous network traffic patterns.

For visualization, we are enhancing the WholeNet viewer [11] to facilitate the analysis of communication pattern anomalies in control system TCP/IP packet traces. Briefly, the WholeNet viewer is a visualization tool that display network log data in two-dimensional gray-scale intensity plots, where the user may assign various hash-function algorithms to be applied to the data in each dimension. Currently, the user may select from source address and/or port, and destination address and/or port for the axes. The viewer reads data in a variety of form, including packet traces in the libpcap format.

The basic concept is to collect data on observables (such as number of packets) indexed by identifiers that can be mapped to coordinate axes (such as IP address). For compactness and scalability, WholeNet hashes the axis coordinates to a limited set of values, and represents intensity of observables on a two-dimensional display, with time as a third dimension. The data display consists of the two-dimensional image with a y-axis histogram on the left and an x-axis histogram below. For coordinate axes, we have found IP source versus IP destination and IP source versus destination port as the most useful displays. The intensity (darkness) of the displayed pixel at any location is proportional to the count of the underlying observable. The user has options to invert (reverse black and white) the image or either histogram, compute logarithm before converting to gray scale, remove the row-wise minimum (a form of background noise removal useful for enhancing horizontal scans), and zooming.



Slot2

	dataSelection	dimension	dataTracking
x	DestAddress	16	global dynamic hash
y	SrcAddress	16	global dynamic hash

FrameDecay	MaxFrames	FrameSkip	FramesByMillisec	FramesByCount
0.0	0	0	0	0

millisecPerFrame 1000

□ loop

□ chartLog

Figure 2. Communication Pattern of a Modbus TCP Server Scan

The analyst controls how the input data records are to be used to create the two-dimensional gray-scale plots for one slot through numerous settable parameters. These parameters define the data source for each dimension and the behavior of the hashing function. There are also settable parameters that control how long a time interval (in data units) is consolidated per displayed frame.

The animation replay controls allow the analyst to set how long to display each frame (not to be confused with the data time interval per frame previously mentioned), as well as functions to pause, step, and back up over interesting sections. For example, we can display frames based on a one-minute data time interval at a rate of four frames per second, effectively 240X real time.

Figure 2 depicts a WholeNet viewer snapshot that shows the source address versus destination address plot before (left pane) and after (right pane) an attack step in which a compromised machine in the process control network performed a scan to identify hosts on the network that included a Modbus TCP server. We hash the addresses in the network so that we can map communication patterns on a 16 by 16 display in the WholeNet viewer. The attack is evident in traffic from a previously unseen source to a range of destinations (horizontal feature), with a symmetric vertical feature corresponding to TCP reset packets.

As shown in Figure 3, the WholeNet viewer display shows the effect of another attack step in which a compromised host performed a port scan. The left pane of the display shows the source address versus destination port pattern before the scan, and the right pane shows the communication pattern during the scan. The scan is evident as a horizontal feature in the right panel, corresponding to traffic across most of the entire hash range of port numbers.

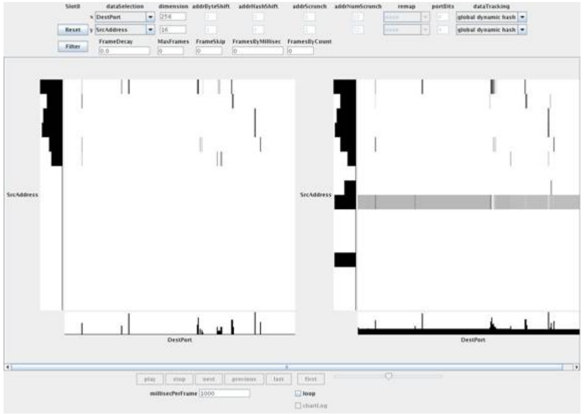
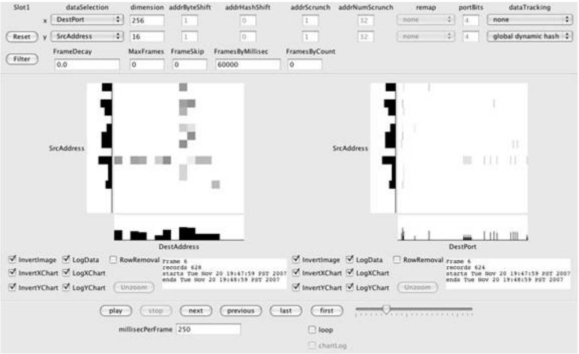


Figure 3. Communication Pattern of a Port Scan

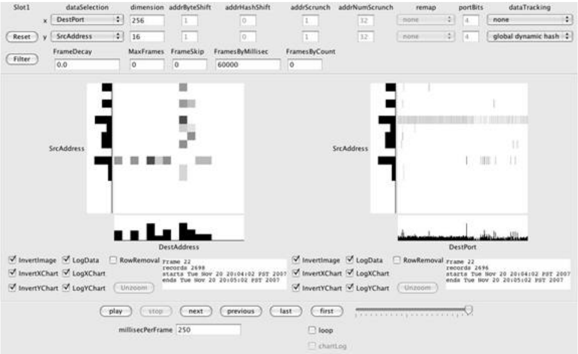


```
Frame 6
records 628
starts Tue Nov 20 19:47:59 PST 2007
ends Tue Nov 20 19:48:59 PST 2007
```

```
Frame 6
records 624
starts Tue Nov 20 19:47:59 PST 2007
ends Tue Nov 20 19:48:59 PST 2007
```

Figure 4. Communication Pattern Before a Port Anomaly

To explore the utility of visualization and pattern analysis of communications for other control system protocols, we have undertaken an initial look at an OPC (`http://www.opcfoundation.org`) trace of 23 minutes duration, from a test system. We observe that when we view the communication patterns at a 60 second frame rate, the communication pattern is comparatively static, and looks like Figure 4. This periodicity was noted by observing the trace at varying temporal frame rates; we will explore analytical techniques such as frequency-based methods to discover such periodicity in an automated fashion. Fast Fourier Transforms (FFTs) may discover temporal frequencies at which traffic patterns tend to repeat. FFT analysis may potentially discover several such frequencies. The results can be used to update the monitoring system's knowledge base to detect anomalous communication patterns. Later in the trace, a new pattern is observed that persists for about two minutes; this is shown in Figure 5. We observe that this is qualitatively similar to the Modbus scan described earlier, for which we have ground truth. For the OPC trace, however, we do not know if the second pattern is anomalous. A longer trace might permit us to determine whether this is a regular pattern that occurs with a longer period than the apparent one-minute pattern, or an anomalous pattern. While this is just an initial exploration on a test system, and a production system may exhibit less regularity, it is indicative that monitoring communication pattern regularity may be useful.



```
Frame 22
records 2698
starts Tue Nov 20 20:04:02 PST 2007
ends Tue Nov 20 20:05:02 PST 2007
```

```
Frame 22
records 2696
starts Tue Nov 20 20:04:02 PST 2007
ends Tue Nov 20 20:05:02 PST 2007
```

Figure 5. Communication Pattern After a Port Anomaly

4. Conclusion

Modern systems in the energy sector are critically dependent on digital controls for the safe and efficient operation of processes in such applications as refining, pipelines, and electric power. These process control systems were formally on isolated networks running application-specific protocols. The pressures of the market increasingly motivate asset owners to adopt commodity platforms and networking protocols in PCSs, sometimes encapsulating legacy protocols that were not designed with security in mind, and to connect PCSs to business systems. As a result, PCSs can benefit from advances in the wider scope of information technology. However, this rapid migration to commodity platforms and standards may expose PCSs to the risk of cyber attack. Because PCSs control physical processes, the consequences of such attacks are not merely economic but can include environmental and safety impacts. The situation is exacerbated because, due to the stringent availability demands of PCSs, enterprise security practices such as system patching are not widely adopted.

Faced with this situation, asset owners are adopting perimeter defenses such as firewalls and switched network topologies in PCSs and Demilitarized Zones

(DMZs) between business and control networks. These preventive security measures are important and effective; however, intrusion monitoring is also essential to ensure that perimeter defenses are not breached or bypassed.

This paper presents a multilayer security architecture that addresses the challenges of PCS monitoring, providing timely and accurate reporting of security-relevant events. This is accomplished through model-based monitoring, whose usefulness has been validated experimentally, at the device, network, and control host levels, leveraging the regularity of network communication patterns and exploiting the special-purpose nature of PCS. Moreover, the architecture employs a hierarchical security incident event management framework to correlate IDS alerts and potentially anomalous events generated by the PCS to achieve situational awareness at multiple levels. To facilitate human analysts to better comprehend network traffic pattern anomalies, we have developed a tool that supports multiple user-customizable views and animation to visualize network packet traces.

Acknowledgment and Disclaimer

This material is based upon work supported by the Department of Energy under Award Number DE-FC26-07NT43314.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes. Using model-based intrusion detection for SCADA networks. In *SCADA Security Scientific Symposium*, Miami Beach, Florida, Jan. 2007.

[2] CNN. Staged cyber attack reveals vulnerability in power grid, Sept. 2007.

[3] D. E. Denning. An intrusion-detection model. *IEEE Transactions on Software Engineering*, SE-13(2):222–232, Feb. 1987.

[4] B. Dutertre. Formal modeling and analysis of the Modbus protocol. In *First Annual IFIP WG 11.10 International Conference on Critical Infrastructure Protection*, Hanover, New Hampshire, Mar. 2007.

[5] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for Unix processes. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 120–128, Oakland, California, May 6–8, 1996.

[6] C. Ko, M. Ruschitzka, and K. Levitt. Execution monitoring of security-critical programs in distributed systems: A specification-based approach. In *IEEE Symposium on Security and Privacy*, pages 175–187, Oakland, California, May 4–7, 1997.

[7] Modbus IDA. Modbus application protocol specification v1.1a, June 4, 2004.

[8] Modbus IDA. Modbus messaging on TCP/IP implementation guide v1.0a, June 4, 2004.

[9] North American Electric Reliability Corporation. Critical infrastructure protection standards.

[10] M. Roesch. Snort: Lightweight intrusion detection for networks. In *Proceedings of LISA '99: 13th Systems Administration Conference*, pages 229–238, Seattle, Washington, Nov. 7–12, 1999.

[11] A. Valdes, M. Fong, and K. Skinner. Data cube indexing of large infosec repositories. In *AusCERT Asia Pacific Information Technology Security Conference*, May 2006.

[12] A. Valdes and K. Skinner. Adaptive, model-based monitoring for cyber attack detection. In H. Debar, L. Me, and F. Wu, editors, *Recent Advances in Intrusion Detection (RAID 2000)*, LNCS, Toulouse, France, Oct. 2000.

[13] A. Valdes and K. Skinner. Probabilistic alert correlation. In W. Lee, L. Mé, and A. Wespi, editors, *Recent Advances in Intrusion Detection (RAID 2001)*, volume 2212 of *LNCS*, pages 54–68, Davis, California, Oct. 10–12, 2001.