A RESPONSE TAXONOMY AND COST MODEL FOR ADVANCED METERING INFRASTRUCTURES

BY

AHMED MOHAMAD FAWAZ

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2013

Urbana, Illinois

Adviser:

Professor William H. Sanders

ABSTRACT

The smart grid is creating the potential for security vulnerabilities due to the deployment of networked devices into the traditional grid. A core component of the smart grid is its advanced metering infrastructures (AMIs), in which a utility communicates and controls smart meters at customer sites. The fine-grained control offered by AMIs increases the risk of cyber-attacks. It is critical to develop cost-sensitive automated response and recovery strategies, because manual management of security incidents in such a large and complex system is impractical. This thesis addresses the challenge of enabling automatic responses to cyber-attacks through two main contributions. First, we introduce and classify an extended set of AMI-specific cyber-incident response actions. Second, we define a cost model for response actions. A cost model is an approach for translating security properties into monetary costs. The cost model is a key element in enabling an automated response engine to make optimal decisions and mitigate cyber incidents. Since AMIs are cyber-physical systems, the cost model accounts for costs due to both the cyber system and the physical system. In particular, the cost model estimates the effects of cyber-responses on the cyber-system and computes the cost due to the loss of cyberservices. The cost model then estimates the consequences of cyber-responses for the distribution grid. The costs incurred within the distribution grid are due to outages, disruptions of the electricity market, and violations of contractual agreements among stakeholders. Finally, we present a realistic implementation of the cost model using ArcGIS for topology generation and GridLAB-D for grid simulation.

إِلَى أُمّي و أَبي و إِخوَتي

To my family, for their love and support.

ACKNOWLEDGMENTS

First and foremost, I thank God, the most Gracious, the most Merciful.

I would like to thank my advisor, Prof. Bill Sanders, for his support. He has been a constant source of encouragement, ideas, and advice. I thank him for pushing me to work harder. I would also like to thank Dr. Robin Berthier, who co-advised me on my work. I thank him for his insight and advice, and for ensuring that I did not stray off-course. This work would not have been possible without him. I also thank Edmond Rogers for the numerous discussions we had, and for sharing his vast practical experience.

Many thanks go to Jenny Applequist for her tremendous help in editing my work. She has never once complained about the insanely short deadlines I have imposed on her, and her assistance will always be appreciated.

The members of the PERFORM group have always provided me with an atmosphere that is both stimulating and enjoyable. I thank Sobir Bazarbayev, Dr. Eric Rozier, Dr. Douglas Eskins, Dr. Sankalp Singh, Ken Keefe, Michael Ford, and the other members of the PERFORM group for their friendship, and for helping me through my transition here.

I thank my dearest friends, and roommates, Ali Houjeij and Mohamad Jammoul, for their company and encouragement.

Last but not least, I am thankful for my wonderful family. I cannot express sufficient gratitude for my parents, my mom Samira and my dad Mohamad, and my brother Kassem and sister Mariam for their endless love, support, and advice. I could not have achieved what I did without their help, and I owe all my success to them.

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000097. 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

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

3G	3rd generation of mobile telecommunications technology
AMI	Advanced metering infrastructure
ART	Attack response tree
ASAI	Average Service Availability Index
CIA	Confidentiality, Integrity, Availability
CIP	Critical infrastructure protection
CPS	Cyberphysical systems
CR	Cell relay
FPGA	Field-programmable gate array
GIS	Geographic information system
GSM	Global System for Mobile Communications
HAN	Home area network
HVAC	Heating, ventilation, and air conditioning
IDS	Intrusion detection system
MDM	Meter data management
NAN	Neighborhood area network
NAP	Neighborhood access point
NERC	North American Electric Reliability Corporation
PLC	Power line communication
PNNL	Pacific Northwest National Laboratory
PMU	Phasor measurement unit

RRE	Response and recovery engine
SAIDI	System Average Interruption Duration Index
SCADA	Supervisory Control and Data Acquisition
SLA	Service level agreement
TCIPG	Trustworthy Cyber Infrastructure for the Power Grid
VPN	Virtual private network
WAN	Wide area network
WiMax	Worldwide Interoperability for Microwave Access

CHAPTER 1

INTRODUCTION

The smart grid promises customers faster and more reliable service. It improves reliability through remote control, quick and automatic detection of blackouts, and accurate state estimation of the power grid using PMUs. Additionally, the smart grid accommodates more customer services, such as real-time pricing, and includes provisions for efficient and safe integration of electrical vehicles. AMIs are a core component of the deployed smart grid.

AMIs are the communication solution for smart meters. Meters in an AMI transmit real-time meter readings to an administrative network and execute received remote service commands. Remote commands can turn off service and even control specific appliances in individual homes; those services are part of demand-response which reduces demand during peak hours to limit environmental and economic impacts. Outages can be accurately detected through AMIs, which decreases recovery time, thus improving customer service and reliability. Moreover, a human meter reader will not be needed, because meters will frequently send usage data through the AMI. Finally, customers will control smart appliances over the Internet with the assistance of real-time pricing information.

AMIs introduce significant security issues, since the processing and communication capabilities of AMI devices allow for a larger attack surface. That attack surface includes 1) the corporate network, 2) the wireless mesh network, 3) the home area network, and 4) meters that are within the reach of customers. Potential threats can be classified according to attack scale, ranging from relatively small-scale activity designed to target specific customers (e.g., to turn off service or specific appliances, such as alarm systems) or steal energy (e.g., through the alteration of meter readings), up to major organized crimes that could target extended geographical regions. Moreover, attacks could target the control commands sent by a utility through the AMI. Additional security issues also arise from the use of wireless solutions for smart meter communication, in particular through the deployment of large mesh networks [1, 2, 3].

Compared to traditional IT systems, AMIs have stringent requirements in terms of quality of service and security guarantees. Those requirements include:

1. Availability: Utility companies should be able to get the latest meter readings and send

out control commands within specific time constraints. Moreover, customers expect the latest pricing to be available.

2. Resilience: AMIs provide a critical service to customers. They must be able to work under extreme conditions and provide the core service of measuring energy consumption even under attack.

3. Fast recovery: In the event of an attack, a compromise, equipment faults, or even blackouts, an AMI should allow fast recovery and restoration of service.

4. Size: In the future, a typical AMI could be larger than any conventional CPS ever built, with millions of nodes in cities; this massive size imposes scalability issues for traditional security solutions.

5. Privacy: There are also privacy concerns specific to AMIs, since the readings and commands sent between the meter and the utility company reveal private information about customers.

Researchers and organizations have made important efforts to promote security solutions for AMIs, such as VPNs, encryption [4], and remote attestation [5]. Those approaches are valuable, but they are not sufficient, mainly because vulnerabilities can always be found in the implementations of protocols and applications, or in human operators who can be tricked into providing access to restricted resources. Moreover, since meters may not have sufficient physical protection, tampering with devices may leak secret keys stored in internal memory and thus could possibly cause security breaches in the network. While recent efforts have started to investigate the role of AMI intrusion detection (e.g., [6, 7]), security administrators must manually respond to incidents. We propose to supplement traditional attack prevention solutions with intrusion tolerance methods such as automatic detection and mitigation approaches.

Automated response and recovery are of critical importance due to 1) the potentially unmanageable volume of alerts and demands for decisions in such a large infrastructure, and 2) the stringent timing and availability requirements of certain power grid functions. In particular, utilities should be able to get the latest meter readings and send out control commands according to specific schedules. Additionally, disruption of services, such as outages, should be detected and addressed with minimum input from human operators. Automatic response to cyberincidents requires a solution that can process input sent by intrusion detection systems, assess the security state of the infrastructure, and select the best response action to mitigate issues in a timely manner.

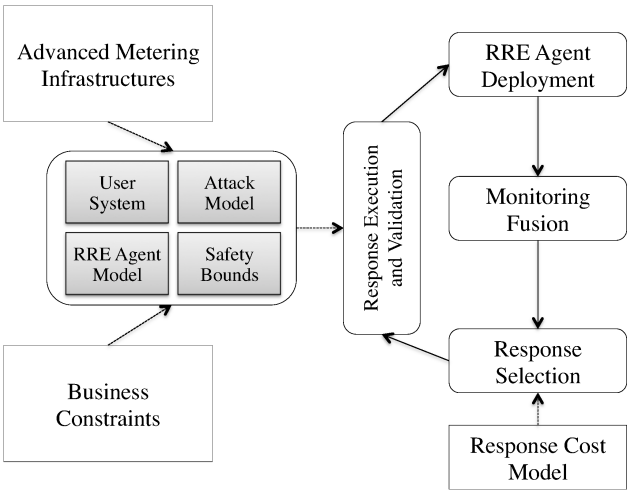

Figure 1.1: Proposed RRE components

We propose to implement the response and recovery engine (RRE), an intrusion tolerance system. RRE determines the security state of the system and detects possible intrusions. In case of an attack, RRE's mission is to contain the attack, maintain quality of service, and recover the system back to a secure state. RRE runs a cost-sensitive response selection algorithm. That is, it decides on containment levels, and the time to start recovery, based on the costs of the possible response strategies. Figure 1.1 shows the interactions among the components of RRE. RRE models AMIs and the business constraints of the utility to generate a user system model, an attack model, and some safety bounds. The user system model describes the system resources and services. The attack model specifies all attack consequences for the system, and the possible response actions for the system. These offline models, along with a response cost, are used to determine the best possible response to deploy. Since the selection algorithm is cost-sensitive, the response strategy is dependent on the accuracy of the cost model. Therefore, it is critical for the cost model to accurately reflect the financial cost a response to the utility of a response, in order to avoid catastrophic outcomes.

This thesis explores the concept of automated cyberincident response for AMIs, proposes a taxonomy of response actions, and designs a cost model for response actions for AMIs. The different chapters address the challenges of automated response as follows:

- Chapter 2 provides background on security, smart grids, and AMI, and reviews and discusses existing automated response frameworks and cost models.

- Chapter 3 introduces an extensive set of AMI-specific response actions through a taxonomy and the identification of key response characteristics.

- Chapter 4 introduces a practical cost model workflow that can translate system logs and cyber-alerts into response costs; provides a solution to generate a dependency graph from an AMI routing topology; and explores the impact of security attributes in the context of an attack.

- Chapter 5 evaluates the cost model in an attack scenario and assesses the potential disruption that a given number of compromised meters could cause.

CHAPTER 2

RELATED WORK

Cybersecurity is the field that studies the security properties of information and information systems. Cybersecurity has evolved over time and has been adapted to support the different uses of information systems, starting from centralized main frames, to the world of ubiquitous mobile users. Cybersecurity also evolved to accommodate increasing levels of threats and risks.

On the other hand, the power grid is being outfitted with smart networked devices. Those devices aim to increase an operator's awareness of the state of the grid, and the reliability of the grid itself. However, the introduction of networked devices may put the grid at risk; cyberattacks targeting the grid could cause outages and even blackouts. Operators should be assisted in the detection and response process, due to the sheer size and complexity of the grid, We propose using automated response and recovery systems. Such a system would detect attacks and suggest cost-sensitive responses to tolerate attacks. Finally, to obtain accurate predictions, it is necessary to have a cost model. Automated response systems cannot decide on optimal actions without an accurate and "complete" cost model.

In section 2.1 of this chapter we present the evolution of cybersecurity and lead the way to the need for intrusion tolerance. We present in section 2.2 the latest work on cost modeling of response actions. We also present in section 2.3 the necessary background information on the smart grid and its operation. Finally we detail the design of AMIs in section 2.4.

2.1 Cybersecurity

2.1.1 Traditional Security

"Security is the state of being free from danger or threat" [8]. Traditionally, computer security has been associated with data security; that is a computing system is "secure" if the data stored within it are "safe." As computers have become a primary support of businesses and industrial processes, the definition has been extended to include the cyberservices being

operated by the computing systems. Thus, cybersecurity is looked at as the physical security of machines, the "well-being" of the services running on the machines, and the protection of stored data.

The situation is well-characterized by the standard rubric to describe security: "cybersecurity is the protection of *confidentiality*, *integrity* and *availability* (CIA) of data and services within a cyberspace" [9], where the terms are defined as:

Confidentiality: Preventing disclosure of information to unauthorized entities.

Integrity: Guarding against improper information modification or destruction.

Availability: The property of being accessible and useable upon demand.

Typically, administrators statically identify and deploy security measures that would protect the CIA properties of resources (data and services) in the system. For example, they would create access control rules for data access, add redundant servers to ensure availability of a Web service, and use signed hashes to ensure that data are not tampered with [10]. However, static methods do not protect systems against attacks that exploit vulnerabilities unknown to security experts, or against misconfigurations or inside jobs.

2.1.2 Evolution of Security

Cyberattacks typically try to violate the security properties of a system. Security events usually "result in unauthorized access to, manipulation of, or impairment to the integrity, confidentiality, or availability of an information system or information stored on or transiting an information system, or unauthorized exfiltration of information stored on or transiting an information system" [11].

Computer systems are an essential part of industrial processes. Moreover, cyberattacks have become more potent and easier to perform; the failure of traditional security measures has turned attacks into a major financial burden for corporations because of losses in production, or loss of availability of services (for e-commerce systems). Institutions have become aware of the risks involved in leaving systems unprotected; thus, it has become imperative to detect and resolve such security events and minimize their damage.

The first step in the evolution of attack prevention involved looking at the attack surface of a system. The attack surface is the set of parts of the system that are accessible to an attacker. The attack surface has three main components: the network, software, and human attack surfaces. The attack surface of a system carries a set of "intrinsic" vulnerabilities that can be exploited by an attacker and may not be prevented [12]. Example include:

Table 2.1: Common Attacker Profiles

Hobbyist	Disgruntled Employee	Insider Aiding Others
Hacktivist	Industrial Espionage	Foreign Espionage
Terrorist	State-Sponsored Attack	

Table 2.2: Generic Threat Matrix [13]

THREAT LEVEL	THREAT PROFILE						
	COMMITMENT			RESOURCES			
	INTENSITY	STEALTH	TIME	TECHNICAL PERSONNEL	KNOWLEDGE		ACCESS
					CYBER	KINETIC	
1	H	H	Yrs-Decades	Hundreds	H	H	H
2	H	H	Yrs-Decades	Tens of Tens	M	H	M
3	H	H	Months-Yrs	Tens of Tens	H	M	M
4	M	H	Weeks-Months	Tens	H	M	M
5	H	M	Weeks-Months	Tens	M	M	M
6	M	M	Weeks-Months	Ones	M	M	L
7	M	M	Months-Yrs	Tens	L	L	L
8	L	L	Days-Weeks	Ones	L	L	L

- Misconfiguration of services communicating with the Internet through open ports
- Zero-day vulnerabilities in the software that implements email, Web, and other services, or
- A socially engineered employee.

Security administrators use risk and attacker profiles combined with knowledge of their attack surface to decide on the level of security investment. The decision is heavily based on the attacker profiles. Table 2.1 shows a list of common attacker profiles. We can imagine that each type of profile has different goals, points of entry into the system, and resources for exploiting the system and concealing the attack. For example, a state-sponsored attack has huge resources, sophisticated capabilities, and specific goals. While a disgruntled employee's goals are different from those of a terrorist, the employee has internal access to the network. The profiles fit into the generic threat table 2.2 proposed by IDART at Sandia National labs [13]. The different profiles are typically modeled using attack graphs [14]. However, more expressive models have been proposed [15].

Online detection of intrusions is needed with the increased risk of attacks. Detection schemes distribute sensors over a system and alert the administrator about possible attacks

or anomalous behavior. There are two types of intrusion detection systems (IDSs), network IDSs (NIDS) and host IDSs (HIDS), distinguished by their location and the type of data they inspect. NIDSs look for packets and flows and are mostly located on routers and firewalls, while HIDSs are located on individual hosts and servers and monitor system calls and logs [16, 10]. Several metrics can measure the performance of intrusion detection systems, such as the false positive rate (FP), false negative rate (FN), true positive rate (TP) and true negative rate (TN). Those rates relate to the number of correctly diagnosed events and missed attacks. Ideally, IDS schemes try to reduce the false rates and increase the true rates. Several detection methods can be used, which have different detection rates.

Misuse detection

In this method, the detection system has a database of attack signatures; an alert is sent when a match occurs. The problems with this method are that new attacks cannot be detected, and the system could suffer performance degradation [10]. This method has a high TP rate, because the signatures are based on expert analysis.

Anomaly-based detection

In this method, the detection system is trained using a "secure" profile of the system [16]. The detection system detects any deviant behavior and generates an alert if, for example, an account is not regularly authenticated from an area. The problems with anomaly-based detection are a high false-positive rate and the possibility of an attacker's poisoning the normal profile by training the detector.

Specification-based detection

In this method, system specifications are studied, and then rules of detection (whitelist behavior) are set. Any deviation from possible legitimate behavior is considered a possible intrusion [17]. This method has the problem that not all systems can be restricted by a set of rules; moreover, it requires expert knowledge to identify rules in the system.

2.1.3 Defense-in-Depth

Different attackers have different entry points to a system, and as an attack progresses, it is important to make sure that the internal structure of the system is as protected as the outer perimeter. For that purpose it is important to implement defenses at all levels in a computing system. This strategy is referred to as *defense-in-depth*. Defense-in-depth is a military strategy; it attempts to inflict maximum damage to an attacker by yielding space

and buying time. In the cyberspace, defense-in-depth is typically implemented by separating the system into layers with firewalls to control access between those layers [18]. Different rules protect the layers from unauthorized access. Moreover, defense-in-depth uses IDSs at the different layers in order to detect intrusions, even if access is controlled because of vulnerabilities in software and firewalls. Other techniques suggested for defense-in-depth include antivirus software, NIDS and HIDS, firewalls, encryption, redundancy, isolation, and moving targets. The critics of defense-in-depth cite the lack of visibility in the network and the decrease of usability (inconvenience) as problems with such defenses.

2.1.4 The Final Frontier

Defense-in-depth's countermeasures might not be enough to thwart an attacker. An attacker could gain access to a network by exploiting vulnerabilities in software and firewalls, or by social-engineering an employee. Thus, an attacker can go undetected, causing damage or stealing information. Attacks could cost an organization millions of dollars and cause disruption of services; thus, it is crucial to build a system that is resilient to attacks. Resilience of a network requires "the ability of the network to provide and maintain an acceptable level of service in the face of various faults and challenges to normal operation" [19]. Resiliency in the cybersecurity space is borrowed from the field of fault-tolerance of computing systems. Typically, designers use redundancy and diversification of components to increase the reliability of a system. Security could benefit from the same ideas of redundancy. Redundancy, compounded with diversification of supply chains and programming languages, could lead to a certain level of attack tolerance. However, diversification is not the silver-bullet solution; it has a high cost and is not implementable by all systems. Thus, we propose attack tolerance through a computer-assisted response and recovery system. The assistance system would help the administrator at several levels:

1. Abstract the security state of the system, thus relieving the administrator from inspecting low-level alerts and logs;

2. Automatically contain an attack or direct the attacker to a honeypot;

3. Suggest optimal responses to ensure tolerance (replication of services, moving targets, etc.); and

4. Automatically generate recovery sequences to restore service and reconfigure the system to avoid future attacks of the same type.

Automated response and recovery systems have to make decisions after assessing the security state of the system. Those decisions require predicting the positive and negative effects of different combinations of attack steps and responses. That element of needing to predict the behavior of multiple entities explains why game theory has often been used in implementing automated reasoning systems for security [20]. Other systems have been proposed by researchers; some use rules to decide on responses [21], while others decide using a local optimization of the costs of possible responses [22, 23, 24].

2.2 Cost Models

The cost model of response actions is the main driver of automated response systems. The accuracy of the cost model deeply affects the decisions made by the response system. A cost model that is not complete or underestimates the real cost might lead to suboptimal or counterproductive actions. We divide past research on cost models into three categories: models based on static costs, models based on parameterized costs with static parameters, and models based on dependency graphs.

2.2.1 Static Cost Models

In the first category, the approaches consist of generating a taxonomy of response actions for general IT systems and then tagging each action with a static cost value [25, 26, 27, 28, 23, 29]. Those costs have to be assigned by system administrators based on their subjective knowledge of the system. The issue with that approach is that it does not capture the system dynamics (i.e., an action that induces changes in a system may affect the costs of subsequent actions). Moreover, requiring administrators to assign cost values is often impractical, and results in inaccuracies.

2.2.2 Parameterized Static Cost Models

In the second category [30, 22], the models decompose the cost of actions into several parameters to better capture how actions may impact the system. Anuar et al. [31] assigned static costs for each parameter and used an analytical hierarchy process to compute impact factors. Luo et al. [32] proposed using static costs that would linearly increase over time. The advantage of these approaches is that the cost model captures more aspects of the actual cost for the utility. However, use of static parameters still does not capture system dynamics.

2.2.3 Dependency-based Cost Models

In the third category, Thoth et al. [33] proposed modeling the system using a dependency graph. The authors dynamically update the availability of the system by propagating in the graph the impact of nodes' becoming unavailable due to response actions. That work was later extended by Jahnke et al. [34] to cover all security properties (CIA) by using three separate graphs (one for each property). Components dependencies across graphs are modeled by adding inter-graph edges between the nodes. Finally, Kheir et al. [24] combined the three graphs into one by labeling the nodes with a vector and used a matrix to model the relation among the different security properties. The importance of that approach is that it models system dynamics to capture the effect of an action on the system. However, the problem is that it still requires considerable effort by system administrators to define parameter values in the graph. Moreover, the output vector, which represents the total effect on the CIA properties, requires additional processing to be used by an automated reasoning system.

2.3 Smart Grid

The power grid in the United States was built during the 1960s. Before then utilities delivered power in isolated islands. When demand increased, especially during extreme cold, local islands failed and could not get help from neighboring grids. By the 1960s the islands had grown, and the decision was made to form three synchronized interconnects (Eastern, Western, and Texas interconnects). The interconnects brought great advantages, but at the risk that a single failure in the grid could cascade into a larger blackout. The New York Blackout of 1965 led to the creation of a utility-managed reliability organization, NERC. The reliability of the grid increased, but after a series of blackouts culminating in the 2003 Northeast blackout, it became obvious that the grid needed a major upgrade to bring it the technological advances of the 21st century. Several goals have been set as part of the efforts to modernize the nation's electricity transmission and distribution system [35]:

1. Increase the use of modern communication technology for control;
2. Use dynamic optimization for resource allocation;
3. Increase the use of distributed and renewable power sources;
4. Use "smart" appliances in residential settings;

5. Develop communication standards to ensure interoperability between devices to be used in the power grid.

Part of the evolution into the smart grid has been the introduction of communication devices into the power grid, as highlighted in the Energy Independence Act. The Act explicitly requires deployment of "smart" technologies and use of digital information and control for operating the grid. The communication infrastructure is intended to increase visibility within the power grid in order to provide better situational awareness and more ways to diagnose and repair faults in the power grid. The smart grid promises higher reliability and efficiency through faster recovery from outages, more efficient allocation of generation, and effective use of renewable energy sources such as wind and solar.

2.3.1 Smart Metering

The smart grid promises greater transparency for customers through the use of smart metering to allow real-time reporting of energy usage and control over power consumption. Smart meters will be replacing traditional meters for all customers. The meters have communication capabilities; the communication protocols are specified in ANSI C12.22. The standard specifies packet structure, addressing, session management, and security modes. The meters are in constant contact with the utility and smart appliances in each household. The communication with the utility allows the utility to send customers real-time energy pricing information. On the other side, meters send outage messages to the utilities. Smart grids have provisions to facilitate efficient charging of electric vehicles, with the goal of charging them during low-demand times in order to avoid straining the grid. Section 2.4 details the different architectures for AMIs.

2.3.1.1 Demand-Response

Demand response is a dynamic mechanism that causes changes in customer power usage in response to price increases, or provides incentives to lower electricity use when market prices are high or when system reliability is jeopardized. Demand-response has typically been exercised by manually calling customers during peak days. However, as smart meters are rolling out, demand-response is being enforced by sending real-time pricing information to customers; the prices will increase during peak demand and forcing customers to hold back on power consumption. A second method involves sending load-shedding messages to customers through the use of smart appliances and protocols such as OpenADR.

2.3.2 Situational Awareness

"Situational awareness is the ability to know what is happening on the grid and to anticipate future problems in order to take effective actions" [36]. Situational awareness has multiple levels, starting from generation and ending at the customers. In today's power grid, utilities rely on phone calls from customers and other utilities to detect outages or critical situations. Through the use of advanced networked sensors, such as smart meters or pole-top devices, information can flow easily among the different entities that run the power grid.

2.3.2.1 Wide Area Monitoring

The 2003 blackout increased interest in wide-area monitoring, which aims to provide system-wide snapshots of the state of the grid. Wide-area monitoring allows operators to detect, in real-time, early signs of system deterioration, starting with deviations from steady-state values of metrics such as power flows, voltage magnitudes, phase angle difference and frequency, and rate of change of frequency [37]. Phasor measurement units (PMUs) are currently used to achieve wide-area monitoring. PMUs use GPS signals to measure the phase difference between the voltages at different buses. PMUs are networked and their measured values are concentrated to be used for wide-area situational awareness. PMUs communication protocol is specified in IEEE C37.118-2. PMUs promise high-quality measurements at an error of 50 μs; however, concerns are being raised at the quality of the currently deployed PMUs, which are showing duplicates and dropped data.

2.3.3 Transmission Automation

The transmission system carries the bulk electricity from generators to distribution systems. The transmission system is made up of transmission lines, buses, transformers and other components that are housed in substations. The smart grid promises more efficient protection and control scheme for the transmission system. Digital relays are used to protect those components; relays are used to clear faults, control breakers, etc. Digital relays communicate over fiber optics to implement advanced coordinate protection schemes; such schemes would try clearing faults and signaling in advance to other relays about the possibility of a fault. Moreover, relays provide relatively low cost protection schemes for transformers of different sizes; such measures increase the lifetime of those components and maintain a record to alert operators for the need to maintenance.

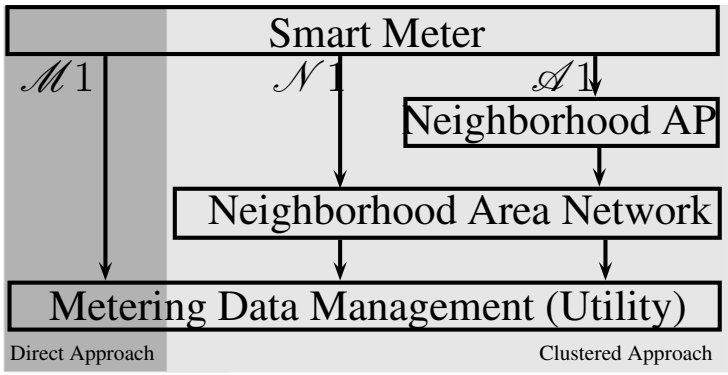

Figure 2.1: Complete view of AMI architectures

2.4 AMI Architecture

The goal of AMIs is to support two-way communications among smart meters, smart appliances, and utilities. Since AMIs can reach huge scales (sometimes more than a million meters) and have to accommodate a variety of environments (i.e., urban, suburban, and rural), several architectures have been proposed for deployment of flexible and cost-efficient communication infrastructure at scale. In this section we enumerate the possible architectures used in AMIs.

2.4.1 Meter Communication

As shown in Figure 2.1, possible options for connecting meters to the utility include two hierarchical approaches and a direct approach. The two approaches have different advantages and disadvantages in terms of security, cost, performance and scalability.

2.4.1.1 Hierarchical Approaches

Hierarchical topologies enable the architecture to scale well with an increasing customer base. In this approach meter traffic is aggregated at several levels before communicating with the head-end. The goal of those approaches is to avoid managing a large number of connections of the direct connection approach. Meters are clustered geographically using one or two aggregation levels to relay communication to and from the utility. Figure 2.1 presents both aggregation levels by showing that a meter can connect to the meter data management (MDM) system inside the utility network through either a collector in the neighborhood area network (NAN) or a second-level neighborhood access point (NAP). A NAP is added to increase scalability. A variety of technologies can be used to deploy the communication

links labeled $\mathscr{N}1$ and $\mathscr{A}1$ in Figure 2.1:

- **Wireless mesh networks** allow for dynamic route generation and route healing. This scheme is the cheapest to implement, and it scales if an optimal placement of collector nodes is used. It is suitable for residential areas where interference between meters is minimal. However, it is prone to a wider class of attacks that can cause availability, integrity, and privacy issues. Examples are physical communication protocols like IEEE 802.15.4 and the proprietary RFLAN [38].

- **Power line communication (PLC)** carries information on power lines by modulating messages to a frequency other than 50Hz. This technology does not require new infrastructure. It is suitable when wireless solutions are not practical, such as in high rises. However, because of the varying impedance, noise, and high attenuation, use of the power line as a channel increases the complexity of the modulator at the meter side [39]. Moreover, since the carrier is the electricity itself, losing a line means losing both power and communication.

- **Wireless Star** uses a collector node that directly connects to each meter. Wireless Star schemes include WiMax or cellular communication that incurs a communication fee per meter. It is suitable for low-density areas, such as rural areas.

- **Private wired networks** are run by utilities; in this approach, a utility would deploy a private network infrastructure (e.g., cable or fiber optic) among meters. This approach is costly, but provides a higher level of security because of the closed nature of the network.

Those hierarchical approaches increase the attack surface by adding collectors, relays, and repeaters to the infrastructure. The attack surface varies depending on the choice of communication technology. For example, wireless communications can be vulnerable to a wide class of attacks, such as jamming, man-in-the-middle, packet injection, and eavesdropping. The choice of technology also impacts the set of response actions available. For instance, it may not be easy to quarantine a node in the case of a wireless mesh network. The clustered approach decreases the visibility of the network; i.e., traffic could be exchanged between meters without passing through the head-end. The decreased visibility would require that extra sensing nodes be placed within the network to detect attacks and failures. Moreover, clustered approaches suffer from performance degradation because of the extra routing levels, and because of the reliance on failure-prone aggregation nodes. However, the scalability benefits and reduced costs are valuable enough to justify adoption of this approach especially in urban areas.

2.4.1.2 Direct Approach

In this scheme, meters directly communicate with the utility through $\mathscr{M}1$ [40]. This approach uses cellular communication (GSM, 3G), WiMax, or leased lines as the communication technology. Direct communication generally offers the security advantage of removing an attack vector by making the network hard to access. However, the scheme adds extra communication cost per meter for the utility and has a scalability issue, especially in dense areas.

2.4.2 HAN Connectivity

For the purpose of enabling demand-response and load shedding, utilities have to gain detailed measurements and control over some of the customer loads. With the advent of smart appliances, utilities have the ability to decrease demand for electricity during peak times by remotely sending price information and even control commands to appliances. Appliances that are most likely to be remotely controlled are high-power-consuming appliances (electric cars, washers, dryers, or HVAC). The communication technology for the home area network (HAN) that connects appliances to the AMI is usually ZigBee. This connectivity brings significant privacy and security issues, since it could enable an adversary to spy on appliance usage, or even to disable specific loads such as a security system. To balance the needs of demand-response applications and privacy and security concerns, several architectures have been proposed. The meter can be the gateway between the HAN and the utility. With a Zigbee radio integrated in the meter, it delivers load-shedding commands from the utility [40]. The HAN gateway can also be connected to the utility through a separate WAN [41], and even use the Internet to receive and send information and commands. The considered architecture leads to the worst-case attack surface.

CHAPTER 3

RESPONSE ACTIONS

The first step towards intrusion tolerance in AMIs involves development of a taxonomy of response actions. The taxonomy suits AMI requirements, such as always preserving the mission of delivering energy and accurately measuring consumption. The taxonomy allows us to construct a set of possible response actions by emphasizing the concept of flexibility. Flexible actions can be tuned to meet a wide variety of requirements and situations. The taxonomy will then guide the development of a practical case study of ways to design flexible actions for an AMI.

In this chapter, we present an extensive set of response actions designed to actively mitigate cyber-incidents in advanced metering infrastructures. The response actions are implementable for all AMI architectures and technologies, with varying levels of effectiveness. Our first effort towards defining those actions was to create a generic taxonomy of actions. Several taxonomies have already been proposed for response actions [29, 25, 42], but most are not suited for AMIs, as they are meant for traditional IT systems. Those traditional set of responses usually targeted computer systems with users and file systems, as opposed to the model within smart meters. Our taxonomy, presented in Figure 3.1, reflects the typical intrusion response process: collecting information, blocking or limiting attacks, recovering from attacks, and performing forensics. This taxonomy has been helpful for 1) exploring the set of possible response actions, and 2) understanding the characteristics of the various actions that are primordial in the definition of cost models. We used the attack trees presented in [43] to map the set of actions to attack techniques and ensure a sufficient coverage of the threat model. The resulting set of actions is presented in Table 3.1.

3.1 Learning Actions

Actions LP1, LP2, LP3, and LA1 (Table 3.1) involve log generation and collection. Those actions are applicable to all architectures; they involve capture of logs generated by the smart meters, relays, and head-ends. Most passive actions do not enact new logging actions, but rely on existing logging mechanisms. LP4–5 are also data-collection learning actions;

Table 3.1: Organized Set of Response Actions for AMIs

Learning	Passive	LP1	Log information
		LP2	Generate reports
		LP3	Generate alarm
		LP4	Profile customers' power usage
		LP5	Passive power bookkeeping
	Active	LA1	Start analysis tools
		LA2	Verify ARP caches
		LA3	Trace connections
		LA4	Enable dormant IDS sensors
		LA5	Detect duplicate nodes
		LA6	Locate routing attacks
		LA7	Request logs
		LA8	Add decoy nodes
Modifying	Blocking	MB1	Block meter
		MB2	Isolate neighborhood
		MB3	Revoke meter keys
		MB4	Restart meter
		MB5	Block connections
		MB6	Limit network access
		MB7	Limit system/service access
		MB8	Enable quarantine
		MB9	Jam attacker
		MB10	Change IP addresses
	Recovery	MR1	Roll back previous responses
		MR2	Merge neighborhood network
		MR3	Distribute attack signature
		MR4	Renew keys of meters/utility
		MR5	Correct C12.22 routing tables
		MR6	Verify meter OS
		MR7	Apply patches
		MR8	Restart meter
		MR9	Replace meter
		MR10	Recover meter readings
		MR11	Recover service state

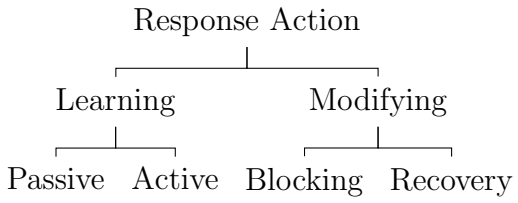

Figure 3.1: Response action taxonomy

however, these collect power measurements as opposed to cyberlogs. They are architecture-independent.

Active-learning actions are mainly used to deploy new sensors or to change sensor configurations in order to collect more activity. Those actions are more efficient when the same activity is visible to multiple sensors. Thus, they are suitable in the case of a shared medium (e.g., wireless or PLC) if meters can be used as sensors. Actions LA1, LA2, LA5, and LA6 are related to verification of routes and detection of routing attacks. If wireless mesh communications are used, then cooperative behavior is needed to verify routes and detect routing attacks (e.g., man-in-the-middle, wormhole, and black-hole attacks) [44, 45]. The responses typically consist of checking routing tables and caches on routers (e.g., cell relays), and sending probe packets to the mesh to verify that routing paths are correct. For example, meters could cooperatively inject multihop traffic in order to verify the integrity of AMI routes. Action LA4 uses meters or utility trucks to enable more IDS sensors, allowing for greater visibility of the mesh network. Finally, action LA8 allows for better understanding of an attacker's skills and knowledge by adding decoy (honeypot) smart meters.

3.2 Modifying Actions

Modifying actions are mostly architecture- and technology-dependent, because their goal is to induce changes in the network.

First, blocking actions aim to limit the access and privileges of a compromised entity in the network. Architectures and communication technologies provide a variety of control functions and granularities. For example, action MB1, "blocking a meter," can be performed by removing the meter from the utility registration list, which is a suitable action for all architectures but may still allow a compromised meter to attack other devices. In the case of a hierarchical topology, a more effective response is to update firewall rules at the level of the collector or the cell relay to block a compromised meter locally. In addition, if a wireless mesh network is used, quarantine of a meter can be performed through updating of routing

information of neighboring meters.

Action MB6 includes rate limiting and is applicable to all architectures. The rate-limiting threshold can vary according to the level at which it is applied in the network topology. The scope of this action also depends on the granularity of the rate-limiting solution. For example, if individual flows for a specific device cannot be isolated, rate limiting at the level of a relay will impact a full neighborhood and will likely affect noncompromised devices. It is also possible to apply rate limiting at the level of the head-end by delaying packet processing for compromised meters.

Recovery actions attempt to return the system to a secure state. They deconstruct operations performed by attackers and require a detailed understanding of the AMI security state. For example, action MR6 checks the integrity of a meter's operating system. Actions MR7–8 can then be used to put the meter back into a secure and working state. MR9 would be used if a recovery is not possible (e.g., the meter has been physically damaged). An important action is MR1, which enables utilities to reverse the effect of one or several response actions. If an action is performed based on incorrect information, or if an attacker is able to take advantage of it, then canceling the action may be necessary. Note that different actions have different rollback levels, ranging from fully reversible (e.g., adding decoy nodes), to irreversible but with removable effects (e.g., blocking a connection), to fully irreversible (e.g., alerting an intruder).

3.3 Response Action Tags

As mentioned in the previous sections, different responses have different intensity levels, different granularities based on the AMI's architecture, and varying rollback capabilities. Those characteristics are used by the cost model when computing the cost of the response, and are used by the response engine when determining if the response is applicable to a certain AMI technology. We tag the different responses with the following characteristics to help us understand cost parameters and to guide the response engine.

1. *Rollback level*: This is an important characteristic to take into consideration when calculating a cost model for each action, since irreversible actions are likely more expensive for utilities than reversible actions are.

2. *Applicability* to specific architectures and communication technologies: For example, the response "merge neighborhood network" (MR2) is applicable only for AMIs that use a clustered architecture.

3. System-level *involvement*: Some actions (such as logging actions (LP1), restarting of meters (MB4), or revocation of keys (MB3)) can be performed locally by a single device; others (including actions that verify routing entries (LA6) or isolate meters (MB1)) require multiple devices to cooperate on a wider scale.

4. *Flexibility*: some actions' intensity can be set. For example, the rate-limiting (MB6) threshold can be tuned dynamically.

5. The *system layer impacted*: Layers include the physical, network, and application layers. It is important to set the layer in order to help with the implementation of the response and determine the privileges it needs.

6. *Manual involvement*: Ranges from none (fully automated action) to some (input required from an operator). This characteristic is critical if an action can have potentially unsafe effects.

CHAPTER 4

COST MODEL

In this chapter we present the cost model for AMI response actions. The cost model fuses the physical state of the distribution grid with the cyberstate of the AMI to compute the financial burden of a response action. We present the method for estimating the cyberstate using dependency graphs that are optimized for AMI services. The cyberstate is the average level of confidentiality, integrity, and availability (CIA) for the AMI services. Finally, we present a method to convert the cyberstate to a financial cost by estimating the consequences of the losses on the power grid.

4.1 Response Cost Model

When accidental failures or malicious events occur, operators of large infrastructures such as AMIs have to take critical decisions in order to recover the system and keep it operating. A diverse set of sensors provides multiple feeds of input raw data that should be converted into information that a human operator can comprehend. The objective of a cost model is to perform that conversion task. At a high level, a cost model takes static input, such as system configurations and system topologies, and dynamic input, such as live feeds of alerts and system logs. The model identifies the cyberphysical state and computes the financial impact due to system changes. System changes include failures, attacks, and security responses. We are particularly interested in the responses; how do we automatically provide quantified decision-making support to operators who are responding to cybersecurity events?

A survey of the literature indicates that availability of services is used as a main metric in computing costs for traditional IT systems. For example, e-commerce systems require high availability to maintain business, and loss of availability is proportional to loss of revenue. However, an AMI is a large cyberphysical system in which the cost of an action is linked to the power grid. Using availability as the cost metric does not represent the actual cost for a utility. Thus we need to quantify the cost due to the cyber system and the interactions between the cyber system and the physical system. This is achieved by our cyberstate conversion process. Additionally, customers play an important role in the system, since they

are directly affected by outages, price updates, and energy delivery services. As a result, we propose a cost model that goes beyond service availability and considers three entities: utilities, customers, and attackers.

When a response action is taken it changes the system state; the notion of response cost can be divided into an operational cost (required to perform the action) and an impact cost (as a result of the action). Indeed, any action requires a preparation phase and leads to a consequence phase, and costs can be estimated for both of those phases. Mathematically, the cost of an action is computed with:

$$C_{Action}(I) = C_{Operation}(I) + C_{Impact}(I) \tag{4.1}$$

where $C_{Operation}$ covers the cost of labor to initiate and run the action, and C_{Impact} is the cost of the impact of the action on the system. The evaluation of those costs requires an accurate prediction of the system's behavior through an in-depth understanding of the internal dynamics. To gain predictive capabilities, one needs to design a dependency model from which state changes of a set of components can be precisely trickled down to other components.

Figure 4.1 presents a high-level framework that describes the dataflows used to compute the total response cost. An important feature of an AMI is that it combines a power infrastructure and a communication infrastructure. We assume that the configuration and topology for both of those infrastructures are static and provided to the model. We generate a dependency graph using the AMI network topology. We use the system logs and IDS sensor alerts to update the intrinsic state of the dependency graph. Using the dependency graph, we compute the security state of the system in terms of impacts on confidentiality, integrity, and availability (CIA) of the different services. We then simulate possible failures in the power grid using the security posture of the cybercomponents. We used a power simulation, GridLAB-D, to compute outage metrics. Finally, we convert the impact on CIA and the magnitude of outages into financial values. Those are presented to an operator or to an automated response system as a response cost.

4.2 Dependency Graph Generation

Our cost model uses a dependency graph of an AMI to evaluate the effect of a response action on the security properties of services in an AMI. The dependency graph is designed to model any system. The dependency graph, $G = (P, E)$, models the dependency relations between the different components $\mathcal{C}$ and services $\mathcal{S}$ in a system, where $P = \mathcal{C} \cup \mathcal{S}$. The edges in the

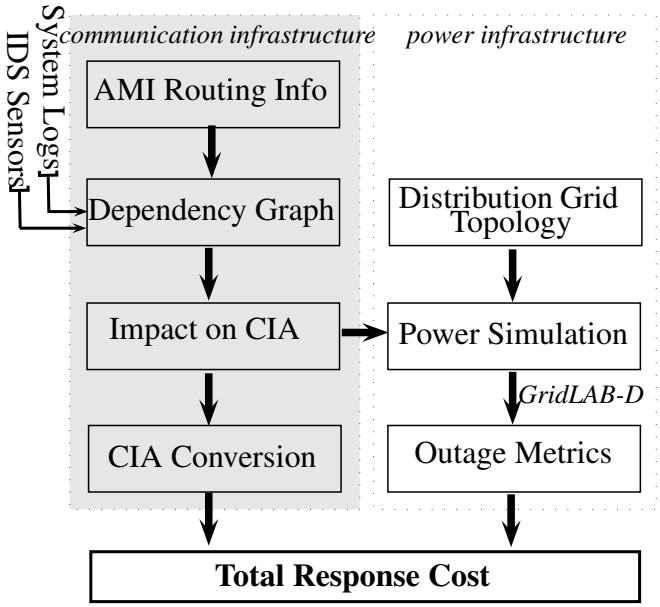

Figure 4.1: Overview of the proposed cost model workflow

graph E model the dependency relation between the components and services. Each studied system has its own set of services and components. Moreover, the relationship between the services and components relates to the system configuration. These relations can be practically modeled by inspecting, protocol specifications, routing information, firewall rules, log files and other traces that can show the dependency relationship between components and services. We define the general method for generating the dependency graphs and also specify the method for creating them for AMIs.

4.2.1 Services

Services in a system are implemented via processes running on hosts. For example in a traditional IT system, those processes could be HTTP services, Kerberos, or an encryption service. Those processes offer the service over the network to other hosts or users. Generally, it is the responsibility of the system administrator to choose the services in a system.

In our case study of an AMI, we consider the following services: a *meter-reading* service, a *real-time pricing information* service, and a *remote commands* service, $\mathcal{S} = \{\mathcal{S}_{mr}, \mathcal{S}_{pi}, \mathcal{S}_{rc}\}$. Figure 4.2 shows a sample dependency graph containing the meter-reading service. More end-to-end services are performed in an AMI, but we focus on the listed subset in $\mathcal{S}$, as they provide a major portion of AMI traffic.

4.2.2 Components

Components are elements of the system that enable a service to function. Components can be divided into 4 classes: data sources, data sinks, cipher elements, and routing elements. The identification of the components of each service defines the dependency relationship in the graph.

In our AMI case study, the components, $\mathcal{C}$, are the meters, C12.19 table entries, C12.22 messages/links, and head-end components used by each service. Specifically, all services in an AMI can be defined as reads and writes to the C12.19 tables and message flows of C12.22 messages. However, different data entries have different semantics. We differentiate among the table entries and define them as separate components in the dependency graph, as shown in Figure 4.2.

- **Meter-reading service** in an AMI generates metering data due to usage, stores the data in C12.19 table entry D^m_{mr}, and sends it to the head-end using C12.22 messages. The data are stored at the head-end's database D^{he}_{mr}.

- **Real-time pricing service** in an AMI writes real-time pricing information received from the head-end's database D^{he}_{pi} through C12.22 messages to the C12.19 table entry D^m_{pi}.

- **Remote commands service** in an AMI writes special values in the C12.19 table entry D^m_{dr}, when specific C12.22 write messages are received from the head-end. That launches special functions that control appliances and turns services on and off remotely.

Finally, some services use the cipher components for encryption and authentication. The basic building components of those operations are keys, ciphers, and hash functions. In our AMI study case, the cipher operations are defined in the C12.22 standard security mechanism [46]. Standard C12.22 security uses the EAX mode for authentication and protects the privacy of the message using a shared *key*. Our dependency graph models the security mode through the keys K_i and cipher $E_{K_i}(m)$. The keys K_i are specific to each meter and are shared with the head-end. The head-end has the following high-level components: 1) a database D^{he}_i, which stores the data for each service; 2) meter keys K_i's; 3) a utility key (pk, sk); and 4) an encryption-decryption engine ED_{eng}, which performs the cipher.

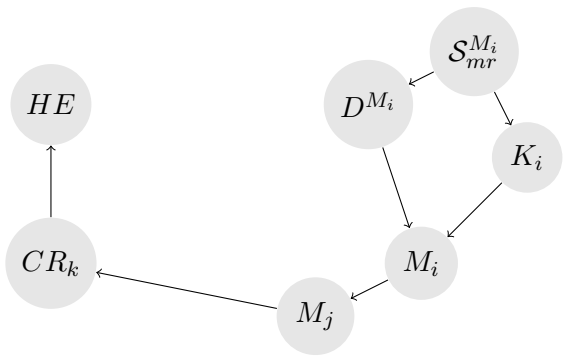

Figure 4.2: Sample dependency graph of an AMI for meter reading service $\mathcal{S}_{mr}$

4.2.3 Graph Generation

The dependency relation $\mathbb{D} = P \times P$ defines the transitive relation between the components and services in G. That is, $(a, b) \in \mathbb{D}$ means "a needs b." $(a, b) \in \mathbb{D}$ is defined as an edge $(a, b) \in E$. We use the routing information, the information flow, and the abstracted service definition discussed in section 4.2.2 to generate those relations. Starting from each service, the graph is generated by tracing the messages from the hosts to the users. For example, in an AMI, a message originates at the head-end, is sent to a meter M_l through a WAN connection to the collector (CR_k), and then is routed using a mesh. The message is verified through use of the key and the cipher $D_{K_l}(m)$ and written to the C12.19 table entry. Thus, a service in an AMI depends on the state of the wireless channel that transmitted the message, the set of meters that relayed the message, the different components in the originating meter, and the state of the head-end.

We start with the set of routes $\mathcal{R}$ in the system, that is, $M_i \rightsquigarrow CR_j \xrightarrow{WAN} Head-end$. We create the routing graph, which represents the backbone of our dependency graph. Then we attach our services and components to the respective hosts. The last step is to connect the dependent services and components. Each meter node contains the services in $\mathcal{S}$ and the dependent C12.22/C12.19 entities we identified. Figure 4.2 shows a sample dependency graph in an AMI.

Each node in the graph is tagged with a vector $V[C, I, A] \in [0, 1]^3$ that represents the current security level of the node in terms of confidentiality, integrity, and availability, with 0 being the least secure state and 1 being the most secure state. The index is interpreted as the current link capacity in the case of availability. For example, if 50% of traffic is dropped through a link, then $V_{link}[A] = 0.5$. However, confidentiality and integrity are binary, with 1 for the secure state and 0 representing the insecure state.

4.2.4 Weight Matrix

Each edge in G is tagged with a function $\mathbb{F}$, $\mathbb{F} : [0,1]^3 \rightarrow [0,1]^3$. The function is a mapping between the security properties (C,I,A) of a component and its dependent. That is, $\mathbb{F}$ defines the effects of a compromise of any of the security properties of a component on its dependent node in G. We implement the function $\mathbb{F}$ as a 3×3 weight dependency matrix $\mathcal{W}$, shown in matrix 4.2.

$$\mathcal{W} = \begin{pmatrix} w_{C,C} & w_{C,I} & w_{C,A} \\ w_{I,C} & w_{I,I} & w_{I,A} \\ w_{A,C} & w_{A,I} & w_{A,A} \end{pmatrix} \tag{4.2}$$

$w_{C,I}$ represents the effect that confidentiality of a component has on the integrity of its dependent node. The weight matrix can be defined for the components we defined in section 4.2.2.

4.2.4.1 Data Components

If the CIA of stored data is affected, then the respective security property of the dependent component is lost. For example, if meter-reading data D_{mr} is not available, then the billing service S_{mr} for the meter is not available. The weight matrix is a diagonal matrix with 1's on the diagonal and 0's elsewhere.

4.2.4.2 Cipher Components

The loss of confidentiality of a key leads to loss of confidentiality of the message, and loss of integrity since the message can be resigned with the compromised key. Loss of confidentiality of a key does not lead however to loss of availability for the dependents. If the integrity of a key is compromised, then the service is using a mismatching key for the security mechanisms. Thus, availability and confidentiality are lost for the end-services. If A is compromised, then C is compromised either because data are sent unencrypted, or because the data are not being pushed through. Those relationships are shown in matrix 4.3.

$$\mathcal{W}_{\mathcal{K}_\rangle} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

Require: $EventSet \leftarrow Logs \cup Alerts$
Require: $G = (A, E)$
1: **for all** $event$ in $EventSet$ **do**
2: $V_{intrinsic}[CIA] \leftarrow Result(event)$
3: **end for**

Figure 4.3: Algorithm initialization

4.2.4.3 Communication Components

The security state of the communication link has a direct effect on the message passing through it. That is, if the link loses availability, the service loses availability; this also holds for confidentiality and integrity. The weight matrix is thus a simple diagonal matrix.

4.3 Security State Computation

The cost model uses the dependency graph to compute the consequences of a response on system dynamics. The model computes a vector $V_{S_i}[CIA]$, which is the security vector of all services in an AMI. Figure 4.3 shows the initialization algorithm. This algorithm requires the set of alerts and logs collected in an AMI. The alerts are mapped to the dependency graph through updating of the intrinsic security vector of all nodes. If an action is a block action, then we have to set the availability of the affected parties. The second step is to propagate the new vector values and compute the per-service security indices, as shown in Figure 4.4. The propagation process starts with a service S_i and finds the dependencies of the services from the meter to the head-end. The algorithm then loops over all sets of (D', E, D), which represents the set of neighbors in the graph. The pair of connected nodes are updated using the update set in line (4). Then, we compute the $V[CIA]$ for S_i by using equation 4.4.

$$V[C] = \min_{i \in depends} V_i[C] \tag{4.4}$$

If a dependent component of a service S_i loses confidentiality or integrity, then the service loses that property. The same argument applies to availability, because the link with the smallest availability acts as the bottleneck in the path. Finally, we compute the total $V[CIA]$ for each service in the AMI in line (9) as the mean of $V[CIA]$. By computing the mean we are computing the fraction of meters that lost confidentiality, integrity and availability for each service.

Require: $G = (A, E)$
Require: $R \leftarrow$ Response Action
1: **for all** S_i in $ServiceSet(A)$ **do**
2: $DependencyPath \leftarrow DependencySet(S_i)$
3: **for all** (D', E, D) in $DependencyPath$ **do**
4: $V_D = \mathcal{W}[E] \times V_{D'}$
5: **end for**
6: $V_{S_i}[.] \leftarrow \min_{D \in DependencySet(S_i)}\{V_D[.]\}$
7: **end for**
8: **for all** S in $ServiceSet$ **do**
9: $V[CIA] \leftarrow \frac{1}{N}\sum_i V_{S_i}[CIA]$
10: **end for**

Figure 4.4: Update algorithm

4.4 CIA Conversion

The third step of the dependency analysis is to convert the security vector $V[CIA]$ into the financial cost incurred by a utility, as described in section 4.1. The cost breakdown for loss of security in the AMI services is highlighted in Table 4.1. The conversion method accounts for the possible consequences of the responses by modeling the effects of an attack on the power grid without explicitly enumerating the possible attack steps. For example, the table shows that the costs of loss of integrity and availability of pricing information and remote commands is due to outages caused by a malicious increase in power demand. The other factors we are considering are due to energy theft (electricity market) and Service Level Agreements (SLAs).

Table 4.1: Attack/Response Action Cost Breakdown

	Integrity	**Availability**	**Confidentiality**
Real-time pricing	Outages		No Cost
Remote Commands	Outages	SLA	Privacy Cost
Meter Readings	Electricity Market	SLA	

4.4.1 Meter-Reading Compromise

Compromising the integrity of meter-reading data could result in receipt of inaccurate metering data by a utility. The meter data are used to bill the customer for power usage; thus, any inaccuracies mean that either the customer is being overcharged, or the utility is losing

revenue. Energy theft is the case of meter data reflecting a value less than the actual one causing revenue loss for the utility. The cost for the utility is shown in equation 4.5.

$$C_{mr} = \Delta T \times \text{estimatedUsageDiff} \times \$/\text{kWh} \times V_{S_{mr}}[I] \tag{4.5}$$

ΔT is the duration of the period in which the response is enabled, and \$/kWh is the cost of power during that period per kWh per unit of time. *EstimatedUsageDiff*, computed in equation 4.6, estimates the value of the power taken during the theft.

$$estimatedUsageDiff = \gamma \times maxUsage \tag{4.6}$$

The value models the aggressiveness of the attack; that is, the more severe the attack the more usage is to be expected. The term γ is an index that reflects the attacker's aggressiveness. Other methods could be used to estimate the amount of energy theft possible; however, such techniques are computationally intensive and would make online use of for the cost model. If the meter readings lost availability, then the utility would miss a ΔT amount of readings. Such a compromise would not lead to a cost for either the utility or the customer, since the data are already stored within the meter and can be transmitted after the AMI is restored to a secure state. However, an SLA penalty would be set because of the unavailability of the service.

4.4.2 Real-time Pricing Information Compromise

Pricing information in an AMI are used for real-time billing. The price is sent to the meter and the customer will pay for the energy used based on that price. In case the pricing information is unavailable, then we assume that the customer is billed based on a flat rate. The utility loses revenue if the flat rate is less than the current price of energy, because the utility is covering the difference in cost. In case the flat rate is more than the actual rate, then the customer is overpaying for energy. The cost conversion algorithm estimates the usage of customers using historic data to compute the revenue loss for the utility.

$$C_{pi} = \Delta T \times \text{historicUsage} \times \Delta\$/\text{kWh} \tag{4.7}$$

If the integrity of pricing information is breached, then we risk a sudden increase in demand as the low prices drive demand. More details on the cost of power demand increase are presented in the next section. Pricing information is public, so there are no concerns about privacy compromise.

4.4.3 Remote Commands Compromise

Service commands remotely control customer service and appliances. This service enables demand-response, which controls prices by varying demand during critical times. The loss of integrity of remote commands allows an adversary to arbitrarily increase power demand in the power grid. The demand increase has two cost repercussions for a utility if the new demand is a significant fraction of the total demand on the transmission grid. The first direct cost is the fact that the increase of demand will require the utility to purchase generation power at a high cost. The indirect cost is incurred by local outages due to severe and persistent demand hikes. Outages are caused when components of the distribution grid (such as transformers, fuses, lines and capacitor banks) are overloaded for long periods of time. Modeling of the distribution grid and the wholesale market is used to predict the costs in the case of the increased demand. The value of increased demand is found using equation 4.8.

$$Demand = \gamma \times maxPower \times V[I] \tag{4.8}$$

γ is the same parameter as in 4.6; it models the effect of the attack. It can be set to depend on the criticality of the location or on the level of pessimism the utility is exercising. Outages require that trucks be rolled out and repair damage. They also cause losses for customers, the cost for customers is highlighted in [47] based on an empirical study.

4.4.4 Cost of Privacy

$V[C]$ is the fraction of private usage data that is compromised. Several papers propose methods for identification of loads that use usage data. Thus, revealing of usage information through loss of confidentiality leads to an invasion of customer privacy. Although it is not easy to put a price on privacy, we have the following suggestion:

- The Californian smart grid law requires utilities to protect usage information. A penalty, similar to NERC CIP violation penalties, could involve a penalty per day per customer for compromised data.
- Recurring privacy compromises might lead customer's to lose confidence and change service providers.

CHAPTER 5

IMPLEMENTATION AND CASE STUDY

In this chapter we discuss the implementation details for the cost model and provide a case study.

5.1 Implementation Details

We now complement the previous chapter by providing lower-level details about the implementation. We start with the details of the dependency graph implementation. We then specify the method for injecting the responses and computing the security vector $V_i[CIA]$. Finally, we convert the security vector to the financial cost incurred by the utility due to the distribution grid.

The response action cost model is designed to run continuously as part of the response engine decision algorithm. The response engine works in an online fashion to help decide on a suitable response during a security incident. As mentioned previously, the model has two phases when computing the cost of a response, updating the security vectors of the dependency model and then simulating the power grid to compute the physical costs.

5.1.1 Dependency Graph

In a practical deployment setting, the dependency graph is generated from a grid topology and a network routing configuration. We wrote a script that takes geographic information system (GIS) data as the sole input and automatically infers an AMI deployment configuration. To test the script implementation, we obtained GIS data from the publicly available parcel tax data of a small town (with a population of 7,282 inhabitants in 2011). We assumed a single meter per parcel, and we located a number of cell relays to cover the GIS region. Using ArcGIS's Python extension, we generated the topology of the deployment. Then the cell relays (CRs) were added to the network to cover the deployed meters allowing 3,000 meters per relay. Using Dijkstra's shortest path algorithm, routes between the meters were

formed to allow each meter to reach the closest CR in the area. Optimal routing can be used to generate the routes since most of the AMI routing algorithms reviewed generally tend to generate optimal routes. Finally, the dependency graph is generated using the algorithm specified in section 4.2. All security vectors are initialized to the secure state of the network, that is $V_i = [1, 1, 1]$, where i represents indices of nodes in the system.

5.1.1.1 Performance Evaluation

We implemented the dependency graph using a data structure called adjacency list to store the state of the different node in the system. Each meter node has 8 components including 3 services, 4 data objects, and 1 cryptographic key, and is tagged with a security vector that holds three floating values (for confidentiality, integrity, and availability). As a result, the memory footprint of the graph is $2N \times (3 \text{ services} + 4 \text{ data} + 1 \text{ key} + 1 \text{ meter}) \times (20\ textbytes)$, where N is the number of meters in the AMI, and the 20 bytes represent the size of the security vector and the overhead due to the graph. The second part of the dependency graph implementation is the update algorithm. The algorithm in Figure 4.4 has a worst complexity $O(|P||E|)$; however, the update algorithm is optimized for the AMI study case. The algorithm updates the security vector per service by traversing the path from the service to the head-end. Since in an AMI the average length of a path is relatively small, the algorithm has an average complexity of $O(|P|)$.

5.1.2 Execution Model

The proposed cost model starts with updating the security state of the system by updating the dependency graph. The update changes the security vectors for affected nodes in the system, and then the new values are propagated throughout the dependency graph. A response $\mathcal{R}$ has to be injected into the dependency graph in order to be evaluated; section 5.1.3 explains the process of injecting a response. After the response has been injected, the update algorithm is run again in order to update the security vectors in the graph. Finally, we have to convert the new values into the financial values using the process described in section 4.4. Implementation details on the financial conversion process due to the power demand in the grid are discussed in section 5.1.4.

5.1.3 Response Injection

The taxonomy of response actions described in Chapter 3 tagged the responses with several parameters. The tags are used to compute the cost of an action. The parameters include response scale, meters involved, and time at which the action is applied. Since an action induces changes in the dependency graph, the security vector $V[CIA]$ gets updated based on the type of action and the expected effects on the system.

The response action taxonomy has two kinds of actions, *learning* actions and *modifying* actions. Learning actions do not change the topology of the network; instead they add traffic to the network in order to gain more information about an event. When a learning action is injected, the availability of a meter or a link is affected because of the traffic needed to get the data. On the other hand, modifying actions can either block or limit an attack, or recover from an insecure state. A limiting response action that blocks a set of meters results in voiding of the availability of the meters, so $V[A] = 0$. Reboot actions cause unavailability for the duration of the reboot process. Recovery actions generally reset the security vector V to $[1, 1, 1]$. However, some actions, such as flashing a new firmware or verification of routes and meters, temporarily cause reduction in availability.

The last step in the communication-side implementation is to update the security vectors. The update algorithm computes the new security vectors in the network. We then convert the CIA indices to the financial cost based on the expected consequences of the action and the actual cost of implementing the action.

5.1.4 Grid Simulation

Parts of the conversion process require predictions of outage frequency and duration. An outage is assumed to be caused by prolonged overloading of parts of the distribution grid. Localized outages require the utility to dispatch crews to repair and restore service, thus inducing cost to the utility. GridLAB-D [48] id used to simulate the grid while the power demand of residential loads is being increased. GridLAB-D is an open-source, agent-based distribution grid simulator developed and maintained by PNNL. The simulator models the topology of the grid including feeders, transformers, capacitor banks, loads and protection devices (reclosers, sectionizer, and fuses). We feed the new security vectors to the GridLAB-D application, which determines the amount of extra demand of power that might occur because of injected response actions. That models the notion of consequences enabled by the action after an attacker executes a particular attack.

The GridLAB-D reliability module models faults in the grid that result from short-circuit

faults such as Single-Line-Ground and Line-Line faults. GridLAB-D allows the user to set the target, timing, and frequency of the fault. However, it does not model faults due to overloading lines in the grid. After trying to increase the power demand we noticed that GridLAB-D notes the overload but does not model faults because of this state. For this purpose of actually injecting faults, we added the demand into loads and record the lines with power flow over the rated level. Then, we inject the faults that would be caused by the sagging of lines, destroyed fuses, or damaged transformers. GridLAB-D simulates the faults and reports outage frequency and sizes and some relevant reliability metrics through the reliability module. The cost for a utility to fix the fault is then estimated using an assumption about the an average cost to repair an outage, $SAIDI \times repairCost(t)$, where *SAIDI* stands for the system average interruption duration index, which shows the average duration of a sustained interruption for a customer during the reporting period. Another important metric used by utility and given as an output in our simulation is the average service availability index (ASAI), which shows the fraction of time that a customer has received power during the reporting period.

5.1.4.1 Performance Evaluation

Our current GridLAB-D simulations are running on a MacBook Pro, with Core i7 2.4 GHz processor and 8 GB of memory; Table 5.1 shows the time it takes to complete a simulation. GridLAB-D uses a single thread to run the simulations. Since several responses are expected to be evaluated and compared during a decision, it is important to compute costs as fast as possible. We note that the results from Table 5.1 could be improved, since work has been done in [49] to accelerate the Gauss-Seidel power flow solver using FPGAs. The authors of the paper achieved 48x-62x speed-up for power flow computation for 200,000 bus system.

Table 5.1: Measured Execution Times of GridLAB-D [49]

# Bus	**Execution Time (sec)**
10,000	2.62
20,000	5.34
50,000	15.32
100,000	32.83
200,000	67.64

As part of the future work, we will be looking to optimize the performance of the simulation. The initial idea is to simulate a large scale power demand situations that can be

stored in a lookup table. The table will define the outage metrics for different values based on power demand and expected demand due to an attack.

5.2 Case Study

In this section, we show how the cost model works in a case study. First, we investigate the level of power consumption needed to drive the grid to a critical state. Then, we use the parameters computed in the first step to study the costs of responding to an attack scenario that compromises the demand-response mechanism in an AMI.

5.2.1 Demand Increase

We simulated the IEEE 37-node test feeder [50] and varied the power consumption of loads in the topology. The simulation verified that the concern that the power supply could be disrupted by an increase in demand is realistic. Moreover, the simulation found the minimum γ to be used in the cost model. We assumed that all loads in the topology were residential loads. Typical residential loads reflect a set of common house appliances that are capable of generating 40 kW of demand. Table 5.2 shows the ratings of some common high-power-demanding appliances. Assuming that demand-response systems typically control heavy loads in order to curtail demand during high demand periods. The maximum possible power injection due to the 37 residential loads is $P_{max} = 1.5$ MW. We simulated the surge of

Table 5.2: Typical Power Consumption in a Household [51]

Appliance	**Rating**
Oven	12 kW
AC	9.2 kW
Heaters	9 kW
Water Boiler	5.5 kW
Dryer	5 kW
Total	**40.7 kW**

demand by increasing residential demand by $\gamma \times 40$ kW. Figure 5.1 shows the current level with varying power demand. We measured the fraction of current feeder to current rating in the main feeder as the demand increased. The results show that it is possible to drive the grid to a critical state with a minimum value of $\gamma = 0.6$.

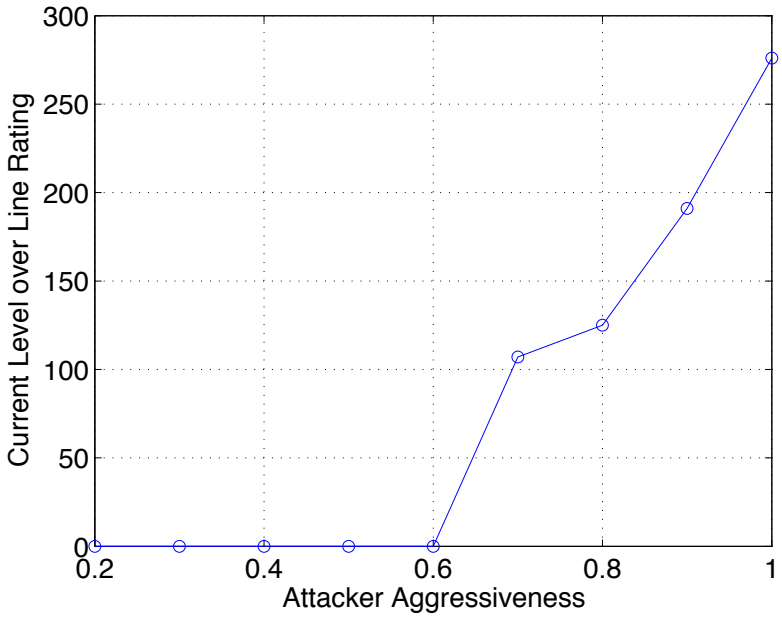

Figure 5.1: Current level with varying power demand

5.2.2 AMI Scenario

We wanted to showcase our cost model for response actions using an AMI under attack. We defined an AMI deployment with the assumptions given in section 4.2. The AMI serves 37 customers, and the meters were placed according to the method suggested in section 5.1. We considered a specific attack scenario and defined two different response actions that would block the attack and recover to a secure state. The cost model computed the cost of the responses using the parameters shown in Table 5.3. Historic usage profiles and price forecasts were used to estimate the flat and peak demand rates, and SLAs were in place to penalize for loss of availability of some AMI services. Finally, the customers were connected to the grid through the IEEE 37-node test feeder topology.

5.2.2.1 Attack Scenario

The attack scenario assumed that a vulnerability in the meters made it possible to replay demand-response commands. The attacker controlled appliances by replaying previously issued commands. The intensity of the attack, the parameter γ, modeled the extent of the adversary's ability to sniff and store messages to be replayed, which is identical to the maximum damage the attacker can inflict.

Table 5.3: Case Study Parameters

Parameter	**Value**	**Parameter**	**Value**
Node location	GIS	Repair cost	\$100/hr
Total nodes	36	T_{patch}	1 day
P_{max}	1.5 MW	ΔT	5 hrs
γ	0.6	Power surge	3 kWh
T_{trans}	18.5 sec	Flat rate	\$4
SLA DR	\$3,000/hr	Peak power cost	\$4.50
SLA MR	\$500/hr	Failure rate	2×10^{-5}

5.2.2.2 Response Actions

The system allows for two possible response actions. The first response, $\mathcal{R}1$, is to re-key the meters in order to invalidate the packets stored by the attacker. The meters are frequently re-keyed at a rate of ΔT, until a patched firmware is sent to all meters. Meters are re-keyed because the attacker will store the new set of messages to use to restore the attack. Then, after T_{patch}, the meters are patched and the re-keying stops. The second response, $\mathcal{R}2$, re-keys all the meters for one time, then blocks the remote commands service. Then, after T_{patch}, the meters are patched and service is restored. The first response does not completely block the remote control service, but instead allows the remote service to run while it changes the AMI keys periodically. The response also risks the possibility of replaying some demand-response messages between the re-keying periods. The other response does not allow for the risk of replaying demand-response messages, but it completely blocks the remote commands service until a firmware fix is available.

5.2.2.3 Response Cost

The cost of $\mathcal{R}1$ is shown in equation 5.1. The re-keying process generates new meter keys at the head-end, and then sends them to the meters. The periodic key change causes a short temporary unavailability of all services during the roll-out time. After the meters have been re-keyed, the attacker can reinitiate the attack and thus cause an outage in the grid. The first part of the equation reflects the cost of loss of availability of real-time pricing information due to the re-keying process, shown in equation 4.7. The second part of the equation ($SLA(All)$) is the SLA penalty due to loss of availability of the AMI service. The

last part of the cost is one due to possible outages, as explained in section 5.1.4.

$$C(\mathcal{R}1) = (N \times 3\ kWh \times 0.5\$/kWh + SLA(All)).\frac{T_{patch}}{T_{trans}} + SAIDI(\gamma).\Delta T.repairCost \tag{5.1}$$

The cost of $\mathcal{R}2$ is shown in equation 5.2. The cost reflects the SLA penalty due to the unavailability of the remote control service of the AMI.

$$C(\mathcal{R}2) = SLA(\mathcal{S}_{rc}) \times T_{patch} \tag{5.2}$$

5.2.3 Numerical Analysis

In this section, we compute the actual costs of the response actions $\mathcal{R}1$ and $\mathcal{R}2$ using the parameters in Table 5.3. We first find SAIDI by injecting faults in the grid and use the result to compute the costs.

5.2.3.1 Fault Injection

We vary the failure rate of injected faults in our simulated grid and compute the different reliability metrics. The increasing rate models the effect of increasing demand in the grid. The results in Figure 5.2 show SAIDI as a function of the failure rate. The number of outages increases as the failure rate increases over a period of a day. The results also show that at a failure rate of 2×10^{-5} SAIDI was about 6.

5.2.3.2 Numerical Results

In those parameters, the SLA penalty for blocking the remote service is high, which is expected because the remote service provides demand-response, which offers major cost relief to the utility. The parameter *repairCost* is also relatively high, because it includes labor hours and rolling of trucks. The time needed to roll out the keys during the re-keying process is estimated as $T_{trans} = N \times 500$ ms.

First, we computed the SAIDI due to $\mathcal{R}1$. Since $\gamma = 0.6$, which represents the first value of demand when the grid is in a critical state, we increased the failure rate to find the lowest failure rate at which the distribution grid begins to show sustainable outages. The study showed that at a failure rate of 2×10^{-5}, the SAIDI was about 6.

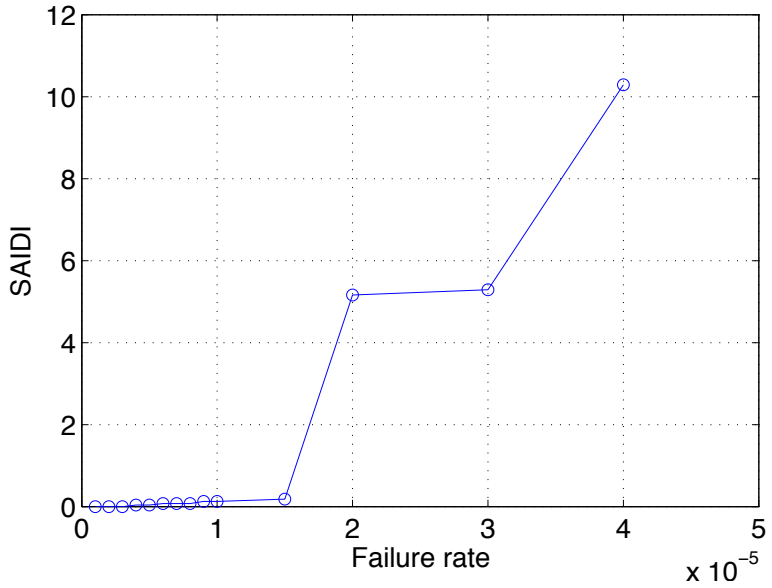

Figure 5.2: SAIDI as failure rate varies

The cost of $\mathcal{R}1$ is $\$3,000$, while the computed cost of $\mathcal{R}2$ is $\$72,000$. Even though $\mathcal{R}1$ has SLA penalties for all services, the penalties are applied for a small period of time that the actual cost diminishes. If ΔT is longer, then the cost to repair outages could have been higher than the SLA penalty for the remote commands service. As a result, based on the value of the parameters the system recommends to apply $\mathcal{R}1$ in case of an attack against the demand-response system.

CHAPTER 6

TCIPG DEMO

For the 2012 TCIPG Annual Industry Workshop [52], we prepared a demo of a comprehensive intrusion-tolerance suite. The demo implemented a preliminary version of a game-theoretic response and recovery engine (RRE) [20]; RRE is a cost-sensitive intrusion response system, and we implemented our AMI-specific response cost model as part of RRE's decision algorithm. RRE uses a specification-based IDS [53] for AMI. Finally, the demo was implemented over TCIPG's Itron smart-metering testbed. The IDS uses logs and C12.22 packets generated by the testbed to generate alerts. RRE uses the alerts to assess the security state of the system and to find optimal cost-sensitive responses.

6.1 AMI Testbed Topology

Our AMI deployment for TCIPG uses 24 Itron OpenWay smart meters and three cell relays; the AMI topology is shown in Figure 6.1. The meters are capable of wireless communication using RFLAN, a proprietary 900 MHz wireless communication protocol designed by Itron. The cell relays are C12.22 routers' which route traffic between the meters and the head-end. Itron's cell relays are equipped with multiple communication ports, such as an RFLAN, a 3G modem, a WiFi access point, and an Ethernet port. The meters form a self-healing wireless mesh network that optimally connects to the nearest cell relay. The cell relay connects the meters and the head-end, as discussed in Section 2.4; the meters communicate with the relay using the RFLAN port and the relay communicates with the head-end using the Ethernet port. We distributed the meters over three floors in the Coordinated Science Laboratory, on the campus of the University of Illinois at Urbana-Champaign. The meters automatically formed a mesh networks on each of the three floors. Each mesh network formed a NAN and connected to a cell relay. The cell relays in our setup connect to the head-end using an Ethernet switch. The head-end is a host that runs several Itron OpenWay collection engine processes; these include Itron's control panel, Itron's C12.22 services, and an Oracle database for storage. Itron's control panel is a Web-based front end that allows control of the AMI configuration.

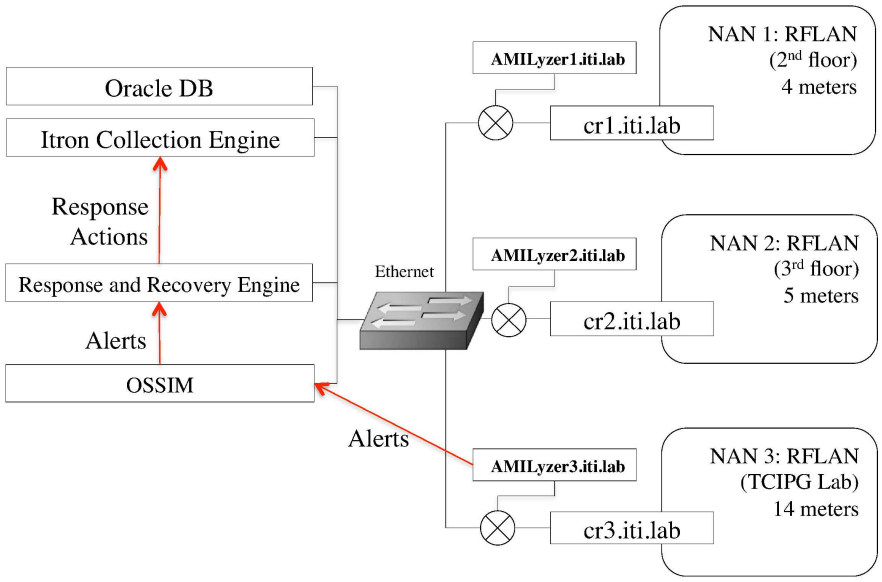

Figure 6.1: Topology of AMI deployed in CSL

We also placed four separate IDS sensor nodes (AMIlyzer) in each NAN. AMIlyzer implements the specification-based IDS, is installed over a beagleboard running Ubuntu. Traffic from each NAN is mirrored using a span port to the beagleboard to be analyzed by the AMIlyzer. AMIlyzer monitors the traffic and detects anomalous behavior as defined in a specification. Alerts generated by AMIlyzer are broadcast and captured by OSSIM, an event manager. We installed RRE on a virtual machine; RRE acquires alerts from OSSIM and continuously assesses the security state of our AMI and computes optimal responses.

Finally, we implemented a front-end, to be used by an administrator, which shows a representation of the system model, and prints possible malicious events and response actions to be deployed to stop an attack and restore the system. Our current implementation assumes that the administrator is in the loop and has to approve all responses.

6.2 Threat Scenario

The demo showcases the detection and response systems that we added to the testbed. First, we preloaded a simplified Attack-Response Tree [20] (ART) to RRE, shown in Figure 6.2. An ART is an extended attack tree that tags each attack step with a set of countermeasures and allows for computation of the probabilistic state of the system using the uncertainties of IDS alerts. An abstract attack goal is the root of the ART. The root goal is achievable through a set of concrete subconsequences. The consequences are connected to parent nodes through AND/OR gates. The logic gates define the precondition Boolean expression needed

to enable higher-level attack consequences. Finally, the lowest-level attack consequences are tagged with a set of observations that are needed to confirm its occurrence. The observations can be IDS alerts, alerts from anti-viruses, or logs. When an observation is captured, the attack consequence is turned into a True. The state of the system is updated through propagation of the changes via the Boolean relations encoded in the ART. Finally, when a countermeasure is enabled, the subtree rooted at the respective attack consequence is set to False. However, as we noted in Section 2.1, IDSes are not accurate; thus, the state computed by updating of the binary state of the tree is not accurate. In order to deal with uncertainties, the observations are tagged with uncertainties; a *naive Bayes binary classifier* uses those uncertainties to compute the probability that the attack has occurred, and thus computes a belief state vector.

We designed our scenario's ART with a service loss as the highest-level attack goal. Service loss can be achieved by 1) injecting disconnect C12.22 packets into the AMI network, or 2) injecting a disconnect job using the AMI Web services. The two possible subconsequences are connected with an OR gate, which means that only one attack consequence is needed to disable service in a meter. Finally, it is possible to inject Web service jobs by connecting to the Web service and sending a C12.22 job. All the low-level attack steps are tagged with observations {Event12,Event13, IDSEvent1}; those are the events we expect from AMIlyzer and the firewall logs. In particular:

- **Event12**: Obtained from any AMIlyzer node
- **Event13**: Obtained from the firewall logs at the collection engine
- **IDSEvent1**: Obtained from any AMIlyzer node

The consequences are also tagged with possible countermeasures, such as blocking C12.22 traffic, if the injection of C12.22 is achieved. As for the unauthorized Web services job injection consequence, we can either block the offender IP, or disable all remote control services. The following is a set of the possible responses:

- Block all Webservices (R1)
- Block unauthorized user (R2)
- Block all meters (R3)
- Reconnect meter (R4)

We implemented responses R1 and R3 using shell scripts that control firewall rules at the collection engine. In effect, we block all incoming traffic to the port used by the Itron Web

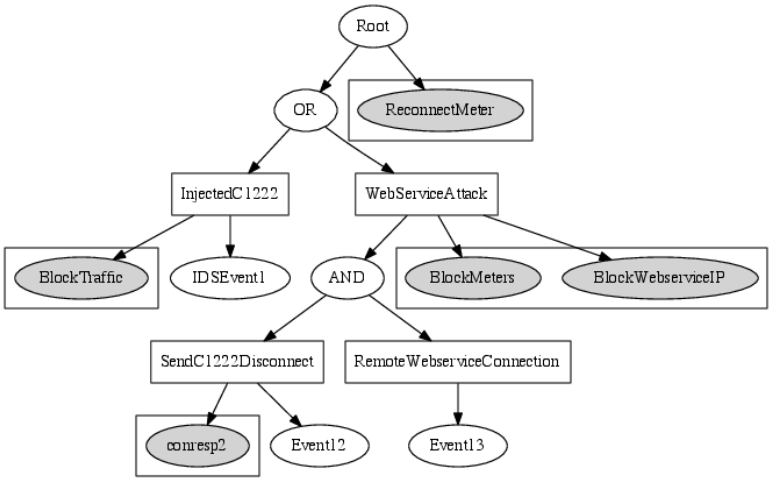

Figure 6.2: Simplified ART for TCIPG demo

services. We implement response R2 by changing some data entries that manage users in the AMI in the Itron database. WE implemented response R4 using a remote script that invokes a Web service that schedules a reconnect job at the collection engine.

6.3 Demo Execution

In our scenario, the attacker, an insider, 1) connects to the Itron Web services and 2) issues an unauthorized meter disconnect order. The combination of the two steps results in the high-level goal of disconnecting service for a customer. We implemented that using a simple C# program that connects to the Web services on the collection engine and calls a meter disconnect function while providing:

- The target meter Electronic Serial Number (ESN)
- Employee login credentials

After the attacker runs the attack script, a C12.22 disconnect message is sent to the target meter. The targeted meter disconnects service for the customer. However, the attack is detected, since it violates the specifications set for AMIlyzer. Thus, an alert is generated stating that an unauthorized, even if valid, disconnect was detected. The alert is captured and stored by OSSIM and pushed to RRE. RRE receives the firewall logs from the collection engine, which reports the Web services logins that occurred when the meter was disconnected. RRE updates the security state of the system and infers that an attack has been detected.

Table 6.1: Costs of Responses

Response Pair	Cost
R1 and R4	37.39
R3 and R4	52.76

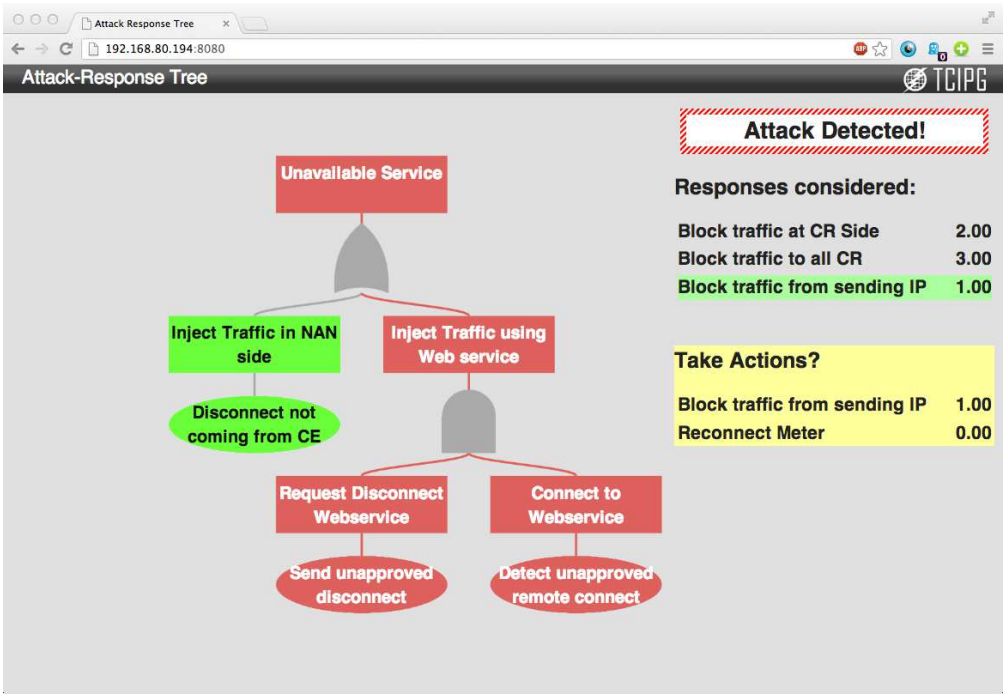

Figure 6.3: RRE front-end showing a set of responses to recover from attack

The response engine in RRE computes the cost of the possible responses and displays the choices to the administrator, as shown in Figure 6.3. The costs of the response pairs as computed by our cost model are shown in Table 6.1. The goal of having an administrator in the loop is to avoid catastrophic outcomes due to RRE. The administrator has to confirm the proposed action by RRE. In our demo, the optimal responses were found to be R2 and R4, where R2 blocks the user from accessing the Web services, and R4 restores service for the customer.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusion

This thesis presented an approach to understanding the role of automated responses for AMIs. We introduced a set of cyber-incident response actions that are suitable for AMIs. The definition of those actions followed a rigorous process that included a review of the possible AMI architectures and communication technologies, the definition of a response taxonomy, and the identification of key response characteristics.

We then proceeded to define a comprehensive approach for cost modeling of response actions. The cost model takes into account the cyber-physical nature of an AMI by integrating system dynamics to capture the potentially significant consequences for the power grid. The approach accounts for cost due to the operation of the action as well as the cost of the action's consequences for the system. We proposed an extended dependency graph that can be used to model any system, and we used it to model an AMI. We then defined methods to convert a response's impact on an AMI to a financial cost to the utility. The methods mainly account for losses due to changes in the physical system, in the form of outages and price fluctuations. Finally, we proposed using a simulator, GridLAB-D, to predict outages in the grid due to power demand increases. The main contribution of this work is that it requires a minimum of subjective input from the utility, as it uses the costs due to the physical grid to specify the importance of services, instead of requiring an administrator to set such a critical parameter manually.

Finally, we implemented the actual automated response system for AMIs on the TCIPG AMI testbed, which contains a hybrid network of real and emulated meters.

7.2 Future Work

We intend to completely automate the process of generating the cost equations using business objectives and costs. Moreover, we plan to implement a robust reliability model for the grid

that accounts for failures due to demand increases. We are also working on improving the practicality of the cost model. Moreover, we plan to use the cost model as a component in a response and recovery engine.

The long-term goal of this research is to build a new response and recovery engine. The engine we plan to build should be scalable, distributed, resilient to attacks and sensor uncertainties, cost-sensitive, modular and adaptive. The goal of the response and recovery engine is to provide tactical and strategic responses based on different threat models and risks. This work will use concepts from control theory, swarm intelligence, machine learning, and game theory. We also plan to devise a similar evaluation framework for intrusion-resilience mechanisms. The goal of the framework is to gain a mature level of evaluation similar to that already achieved in the fault tolerance performance evaluation field.

REFERENCES

[1] M. Al-Shurman, S.-M. Yoo, and S. Park, "Black hole attack in mobile ad hoc networks," in *Proceedings of the 42nd annual Southeast regional conference*, ser. ACM-SE 42. New York, NY, USA: ACM, 2004. [Online]. Available: http://doi.acm.org/10.1145/986537.986560 pp. 96–97.

[2] L. Buttyan and J.-P. Hubaux, *Security and Cooperation in Wireless Networks*. Cambridge University Press, 2007.

[3] Y.-C. Hu, A. Perrig, and D. Johnson, "Wormhole attacks in wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 24, no. 2, pp. 370–380, 2006.

[4] A. Hahn and M. Govindarasu, "Cyber attack exposure evaluation framework for the smart grid," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 835–843, 2011.

[5] M. LeMay and C. Gunter, "Cumulative attestation kernels for embedded systems," *Smart Grid, IEEE Transactions on*, vol. 3, no. 2, pp. 744–760, 2012.

[6] R. Berthier, W. Sanders, and H. Khurana, "Intrusion detection for advanced metering infrastructures: Requirements and architectural directions," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, 2010, pp. 350–355.

[7] Y. Zhang, L. Wang, W. Sun, R. Green, and M. Alam, "Distributed Intrusion Detection System in a Multi-Layer Network Architecture of Smart Grids," *Smart Grid, IEEE Transactions on*, vol. 2, no. 4, pp. 796–808, 2011.

[8] "Merriam-Webster's Dictionary," Mar 2013. [Online]. Available: http://www.merriam-webster.com/dictionary/security

[9] "Presidential Policy Directive – Critical Infrastructure Security and Resilience," PPD-21 (2013).

[10] S. W. Smith and J. Marchesini, *The Craft of System Security*. Addison-Wesley, 2008.

[11] "Homeland Security Act of 2002," Title II, SEC. 242 (2002).

[12] S. Northcutt, "The attack surface problem," 2011. [Online]. Available: http://www.sans.edu/research/security-laboratory/article/did-attack-surface

[13] D. P. Duggan, S. R. Thomas, C. K. Veitch, and L. Woodard, "Categorizing threat building and using a generic threat matrix," Sandia National Labs, NM, Tech. Rep. SAND2007-5791, September 2007.

[14] O. M. Sheyner, "Scenario graphs and attack graphs," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, USA, 2004, aAI3126929.

[15] E. Lemay, M. Ford, K. Keefe, W. Sanders, and C. Muehrcke, "Model-based Security Metrics Using ADversary VIew Security Evaluation (ADVISE)," in *Quantitative Evaluation of Systems (QEST), 2011 Eighth International Conference on*, Sept. 2011, pp. 191–200.

[16] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[17] P. Uppuluri and R. Sekar, "Experiences with specification-based intrusion detection," in *Recent Advances in Intrusion Detection*. Springer, 2001, pp. 172–189.

[18] R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, and R. Cunningham, "Validating and Restoring Defense in Depth Using Attack Graphs," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, Oct. 2006, pp. 1–10.

[19] "Energy Independence and Security Act of 2007," 42 U.S.C. § 17381 (2007).

[20] S. Zonouz, H. Khurana, W. Sanders, and T. Yardley, "RRE: A game-theoretic intrusion Response and Recovery Engine," in *Dependable Systems Networks, 2009. DSN '09. IEEE/IFIP International Conference on*, 2009, pp. 439–448.

[21] Y. Qian, D. Tipper, P. Krishnamurthy, and J. Joshi, *Intrusion Response Systems: A Survey*, 1st ed. Morgan Kaufmann, 2007, ch. 13, pp. 377–412.

[22] C. Strasburg, N. Stakhanova, S. Basu, and J. Wong, "The methodology for evaluating response cost for intrusion response systems," Iowa State University, Tech. Rep. 08-12, 2008.

[23] B. Foo, Y.-S. Wu, Y.-C. Mao, S. Bagchi, and E. Spafford, "ADEPTS: Adaptive intrusion response using attack graphs in an e-commerce environment," in *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*, 2005, pp. 508–517.

[24] N. Kheir, H. Debar, N. Cuppens-Boulahia, F. Cuppens, and J. Viinikka, "Cost evaluation for intrusion response using dependency graphs," in *Network and Service Security, 2009. N2S '09. International Conference on*, 2009, pp. 1–6.

[25] S. Tanachaiwiwat, K. Hwang, and Y. Chen, "Adaptive intrusion response to minimize risk over multiple network attacks," *ACM Trans on Information and System Security*, vol. 19, pp. 1–30, 2002.

[26] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, and S. Dubus, "Risk-aware framework for activating and deactivating policy-based response," in *Proceedings of the 4th International Conference on Network and System Security (NSS)*, Sept. 2010, pp. 207–215.

[27] Y. Wu and S. Liu, "A cost-sensitive method for distributed intrusion response," in *Proceedings of the 12th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, April 2008, pp. 760–764.

[28] Z. Zhang, X. Lin, and P.-H. Ho, "Measuring intrusion impacts for rational response: A state-based approach," in *Proceedings of the Second International Conference on Communications and Networking in China (CHINACOM)*, Aug. 2007, pp. 317–321.

[29] W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok, "Toward cost-sensitive modeling for intrusion detection and response," *Journal of Computer Security*, vol. 10, no. 1-2, pp. 5–22, 2002.

[30] Z. Zhang, P.-H. Ho, and L. He, "Measuring IDS-estimated attack impacts for rational incident response: A decision theoretic approach," *Computer Security*, vol. 28, no. 7, pp. 605–614, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404809000273

[31] N. Anuar, S. Furnell, M. Papadaki, and N. Clarke, "A risk index model for security incident prioritisation," in *Proceedings of the 9th Australian Information Security Management Conference (AISM)*, 2011, pp. 25–39.

[32] Y. Luo, F. Szidarovszky, Y. Al-Nashif, and S. Hariri, "A game theory based risk and impact analysis method for intrusion defense systems," in *Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications (AICCSA)*, May 2009, pp. 975–982.

[33] T. Toth and C. Kruegel, "Evaluating the impact of automated intrusion response mechanisms," in *Proceedings of the 18th Annual Computer Security Applications Conference*, 2002, pp. 301–310.

[34] M. Jahnke, C. Thul, and P. Martini, "Graph based metrics for intrusion response measures in computer networks," in *Proceedings of the 32nd IEEE Conference on Local Computer Networks (LCN)*, Oct. 2007, pp. 1035–1042.

[35] "2011 Roadmap to Achieve Energy Delivery Systems Cybersecurity," http://energy.gov/node/295585, Department of Energy.

[36] "Enhancing Situational Awareness," http://nyssmartgrid.com/innovation-highlights/enhancing-situational-awareness/, accessed: 03/01/2013.

[37] V. Terzija, G. Valverde, D. Cai, P. Regulski, V. Madani, J. Fitch, S. Skok, M. Begovic, and A. Phadke, "Wide-Area Monitoring, Protection, and Control of Future Electric Power Networks," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 80–93, Jan. 2011.

[38] G. Picard, H. Van Wyk, F. Monier, J. Bartier, and A. I. Clave, "Metering RFLAN protocol and cell/node utilization and management," Patent WO2 008 033 514, 2008.

[39] N. Pavlidou, A. Han Vinck, J. Yazdani, and B. Honary, "Power line communications: State of the art and future trends," *IEEE Communications Magazine*, vol. 41, no. 4, pp. 34–40, April 2003.

[40] CEN and CENELEC, "Functional reference architecture for communications in smart metering systems," European Committee for Electrotechnical Standardizations, Tech. Rep. CEN/CLC/ETSI/TR50572, December 2011.

[41] S. Ravens, "Great Britain smart meter infrastructure: Analysis of potential architectures," White Paper, Dec 2010. [Online]. Available: http://www.datamonitor.cloud.ipdgroup.com/detail/OI00034-006

[42] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," *International Journal of Information and Computer Security*, vol. 1, no. 1, pp. 169–184, Jan. 2007.

[43] D. Grochocki, J. H. Huh, R. Berthier, R. Bobba, W. H. Sanders, A. A. Cardenas, and J. G. Jetcheva, "AMI threats, intrusion detection requirements and deployment recommendations," in *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on*, 2012, pp. 395–400.

[44] S. Marti, T. Giuli, K. Lai, M. Baker et al., "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, 2000, pp. 255–265.

[45] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks," in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2005, pp. 49–63.

[46] "Protocol specification for interfacing to data communication networks," ANSI, Tech. Rep. C12.22-2008, Jan. 2009.

[47] A. Chowdhury, T. Mielnik, L. Lawion, M. Sullivan, and A. Katz, "Reliability worth assessment in electric power delivery systems," in *Power Engineering Society General Meeting, 2004. IEEE*, June 2004, pp. 654–660.

[48] "GridLAB-D," 2012. [Online]. Available: http://www.gridlabd.org/

[49] J.-H. Byun, A. Ravindran, A. Mukherjee, B. Joshi, and D. Chassin, "Accelerating the Gauss-Seidel power flow solver on a high performance reconfigurable computer," in *Field Programmable Custom Computing Machines, 2009. FCCM '09. 17th IEEE Symposium on*, April 2009, pp. 227 –230.

[50] W. Kersting, "Radial distribution test feeders," in *Power Engineering Society Winter Meeting, 2001. IEEE*, vol. 2, 2001, pp. 908–912.

[51] "Estimating appliance and home electronic energy use," 2012. [Online]. Available: http://energy.gov/energysaver/articles/estimating-appliance-and-home-electronic-energy-use

[52] "2012 TCIPG Annual Industry Workshop," Oct 2012. [Online]. Available: http://tcipg.org/news/2012-tcipg-annual-industry-workshop

[53] R. Berthier and W. Sanders, "Specification-based intrusion detection for advanced metering infrastructures," in *Dependable Computing (PRDC), 2011 IEEE 17th Pacific Rim International Symposium on*, Dec., pp. 184–193.