A Risk Assessment Tool for Advanced Metering Infrastructures

Tawfeeq Shawly, Jun Liu, Nathan Burow, Saurabh Bagchi
School of Electrical and Computer Engineering,
Department of Computer Science, and CERIAS
Purdue University
Email: {tshawly,liu1234,nburow,sbagchi}@purdue.edu

Robin Berthier, Rakesh B. Bobba
Information Trust Institute
University of Illinois at Urbana-Champaign
Email: {rgb,rbobba}@illinois.edu

***Abstract*—The growing reliance on Cyber technology for Smart Grid and other Cyber-Physical Systems (CPS) applications increases high assurance challenges. Advanced Metering Infrastructure (AMI) is a critical part of smart grid and failure to address security risks in it might disrupt utilities in their mission to deliver power reliably. The objective of this paper is to find mitigation actions for attacks that might be launched against AMI. The paper introduces a tool called SecAMI that calculates the relationship between the speed at which an attack spreads, the speed at which it can be detected, and the consequences on the availability of the AMI. The results from SecAMI can be used as a performance metric to significantly improve the development and the deployment of Intrusion Detection Systems (IDSs) in an AMI environment. Experimental results with an IDS called Amilyzer show that centralized IDS might not work efficiently in scalable systems comparing with distributed IDS in term of detection time, especially with a Distributed Denial of Service (DDoS) attack or remote disconnects on the AMI.**

I. Introduction

A Cyber-Physical System (CPS) is any physical system that is controlled by a computer or network of computers. A prominent example of a CPS is the "smart grid," which among other use cases refers to the use of information on current power demands, for example to determine where to generate power, by how much, how to distribute it, and how to identify utility customers that are willing to use less power during peak demand periods.

The component of the smart grid that will directly impact most people's lives (as opposed to power intensive businesses, or the utility backbone itself) is the Advanced Metering Infrastructure (AMI), sometimes referred to as "smart meters." These meters can provide readings to power companies on demand or be read off remotely. They can also serve as the targets of commands whereby the meters are connected to "smart devices" in the house so that, for instance, the thermostat can be altered, or some appliances can be shut off during peak demand. They can also be used to centrally disconnect a house from the utility network, e.g., for load shedding, or if the customer is in arrears on payments, etc.

Smart meters are typically deployed in a network over a neighborhood; such a group is commonly referred to as a Neighborhood Area Network (NAN). NANs are mesh networks, wherein each node can communicate with a given subset of nodes (based on distance, line of sight, and other factors). In our model, each smart meter also serves as a router, connecting all the nodes it can communicate with. In each NAN there is a Data Concentrator Unit (DCU), which is the central node that gathers data from the smart meters in the NAN and reports back to the utility company. It is also the DCU that has the ability to send remote disconnect commands from the utility to any smart meter.

While smart meters and NANs have many potential advantages, they can also become a double-edged sword by introducing new vulnerabilities into the utility network. If a malicious individual gains control over the DCU, she could have the power to manipulate power loads by changing the data sent to the utility, which might cause brownouts or blackouts [1]. More directly, an attacker who gains control of the DCU has the ability to disconnect the entire NAN. The National Electric Sector Cybersecurity Organization Resource's (NESCOR) report from September 2013, identified a failure scenario in which a malicious individual could launch a mass disconnect [2]. This scenario, AMI.27, considers that a smart meter could be stolen from a vendor and reverse engineered, leading to the identification of a remotely exploitable vulnerability to gain control of a meter.

This paper introduces SecAMI, a simulation tool at the NAN level that calculates the relationship between the speed at which an attack spread, the speed at which it can be detected, and the consequences for the number of meters connected in the NAN. The results from SecAMI can be used as a performance metric to significantly improve the development and the deployment of Intrusion Detection Systems (IDSs) in an AMI environment. SecAMI helps us to answer questions such as which failure scenarios are manageable in terms of network availability as a consequence of detection and response timing and which are not. Further it provides an ability to game out "what if" scenarios with different timing parameters in an efficient way. SecAMI does this by allowing the user to simulate a variety of network topologies and different attack scenarios with varying abilities and goals for the attackers.

SecAMI can simulate any attack that can be modeled as follows. The attacker starts at one of the meters, and has an attack vector that allows her to spread to all of that node's neighbors. Based on the time to compromise a neighbor, the

time to detect that a node has been compromised, and the time to communicate between nodes as inputs, SecAMI plays out the attack and ultimately reports what percentage of the NAN is still connected to the DCU. The attack ends once the DCU has been compromised, or all the compromised nodes have been dealt with according to a chosen response policies.

Two examples of such attacks are a control flow attack and a data flow attack. In a control flow attack, the attacker explores the network, taking control of each visited node, until she reaches the DCU and can issue a mass disconnect. In the data flow attack, the attacker injects false data about the load at each meter into the network, eventually causing a failure [1]. The question that arises for each kind of attack is how quickly can it spread through a network of meters. There has also been some work in identifying response actions [3] when an attack is detected. So a related question is how quickly must an attack be detected and a response action put in place for certain requirements to be met. A natural requirement is that a certain part of the network be reachable from the DCU.

SecAMI achieves those two goals: it quantifies how quickly an attack spreads, and how quickly a response must be implemented in order to maintain a given level of network availability. Additionally, it allows us to examine the effect of a distributed vs. centralized IDS on network survivability. It models an attack scenario, a response technique, the network topology, and the mechanism of spread through the network and computes the portion of the network that is still reachable from the DCU.

We also conducted a case study utilizing the Trustworthy Cyber Infrastructure for the Power Grid (TCIPG) test bed at the University of Illinois at Urbana-Champaign (UIUC) [3]. We used the TCIPG test bed to measure transmission times in the network, and how long it takes to disconnect a node. Additionally, we obtained the amount of time it takes Amilyzer, an IDS that looks at network traffic, to determine that a packet breaks one of its rules and raise an alert. The results were then used as inputs to SecAMI, thus infusing a degree of realism in the configuration parameters. We have open sourced our tool, SecAMI [9], so that smart grid operators can answer "what if" questions that will help them provision and protect smart grids.

The rest of the paper is organized as follows. We discuss some of the related works in Section II. We describe the design principles and objectives of SecAMI in Section III. We discuss how we used the UIUC TCIPG test bed in Section IV. The evaluation methodology is discussed Section V, followed by the results in Section VI. Finally, Section VII contains our conclusions and suggestions for future work.

II. Related Works

In "Who Controls The Off Switch" [1], R. Andersen raises concerns about the possibility that the remote connect/disconnect capability of smart meters might be abused by a malicious attack and causes a large-scale blackout. Here we concentrate on how to prevent such a scenario.

We need an efficient Intrusion Detection System (IDS) with an optimal distribution of sensors in an AMI. Efficiency of any IDS can be determined by the detection time and how fast it is compared to attack time. Therefore, we care about detection time to know what should be the optimal distribution of sensors in a NAN that guarantees to detect an attack leaving the highest possible percentage of AMI alive.

Grochocki et al. [4] proposes a scheme in which the IDS sensors can be embedded in smart meters. The traffic monitoring coverage of this scheme is wide compared to the traditional centralized system that uses only the utility server as an IDS. Consequently, attacks on meters that use these nodes as routers to reach the DCU will be detected much faster. In [5], Shin et al. tried to optimize this scheme by minimizing the number of monitoring meters that give a complete monitoring coverage for all other smart meters and their communications. Additionally, they propose a mechanism to allow recovery of collided packets that are subject to monitoring.

Temple et al. [6] show that when the process of remote disconnects is implemented with a random delay on the order of hours, the utility server would gain enough time to react to multiple cyber-physical attacks. We expand on this work by investigating exactly how long this delay needs to be relative to the time it takes an attacker to issue the disconnect commands. We also quantify the "goodness" of our result in terms of the percentage of the network that is still connected after an attack.

Policy-based approach provides a pragmatic solution for many CPS challenges [7]. Berthier et al. [3] presented Amilyzer, a centralized specification-based intrusion detection system that leverages NESCOR failure scenarios [1] to define a comprehensive set of security policy rules.

III. Design

Our goal is to determine how quickly an attack can spread through a NAN. This is difficult because we are dealing with a network of embedded systems with less processing power than the computers in traditional networks that have been extensively studied. Furthermore, the fact that the mesh network does not have dedicated routers changes the timing of data transmission. Consequently, a new tool built around the characteristics of the NAN networks is needed.

The opposite question is also interesting: how quickly must an IDS respond to an attack? We evaluate this based on how much of the network is still reachable from the DCU after an attack in order to see at a high level how the response and attack times interact.

In addition to how quickly an IDS should respond to an attack, there is a question of what response to choose. We consider two different policies. The first is to disconnect the node from the network. This also disconnects any node whose sole path to the DCU goes through the disconnected node. In effect, the network reconfigures around the change if possible. The second possible response, which assumes encrypted network traffic, is to change the network key and not send the new one to the compromised node. This way,

all traffic from the compromised node would be disregarded because it would not decrypt properly.

Beyond attack and response times, we also look at the effect of where the attacker starts. We start the attack both close to the DCU and far from it to show the effect of the starting point on network survivability. In general, the attacker will not know where the DCU is, and can be assumed to be equally likely to start at any meter in the NAN. Consequently, we report network survivability averaged over the attacker starting at each meter for each of our experiments on attack and response time.

A significant advantage of SecAMI over existing simulators and traditional attack graph approaches is to allow the user to focus on the key characteristics of the attack and defense, such as how the attack spreads and at what rate, abstracting out many of the lower level details. Attack graphs are not useful in this context because meters are all homogenous and so subject to the same access patterns and vulnerabilities. The attacker can move to any node connected to the compromised node. Consequently, attack graphs add no more information than the network topology.

A. Attack Model

SecAMI is agnostic to the motivations of the attacker and how the attack is implemented, focusing instead on how the attack spreads and the relevant timing information. The attacker takes some amount of time, referred to as the attack time, to compromise a node. After that time, the attacker controls that node and can use it to launch new attacks on other nodes. Specifically, SecAMI assumes the attacker attacks all the neighboring nodes. These ongoing attacks depend on the originating node remaining in the attacker's control. If the attacker loses control of the originating node due to a response from the IDS, the attack fails. The attack continues until either the DCU is compromised or the attacker controls all the meters in the NAN (note, this almost never occurs without the attacker also controlling the DCU, but it is theoretically possible).

B. Network Topology

SecAMI can be run on any network the user specifies, with the convention that the DCU is at node 0. For our experiments, we observed that the NAN is a mesh network, most of which follow a power law distribution wherein most nodes have few neighbors, but a few key nodes have many neighbors [8]. We generated example networks with 50 - 150 nodes, and with 2, 6, or 10 as the maximum number of connections per node.

IV. Trace-Based Simulation

To insure that our simulation results are as close to reality as possible, we used the TCIPG test bed to measure the network parameters of interest. We measured the time to communicate with a meter, the time to disconnect a meter, and the performance of an IDS called Amilyzer. The communication time was used in our general simulation, the time to disconnect a meter was used as a proxy for attack time in conjunction with Amilyzer as a case study to see how vulnerable existing NANs might be.

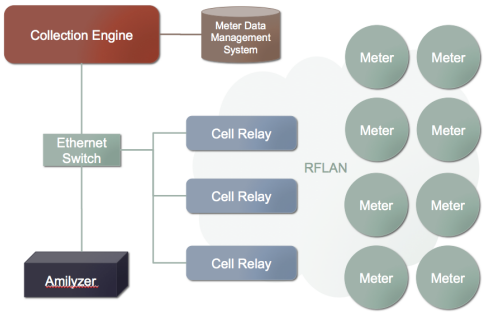

Fig. 1. TCIPG AMI test bed used to derive parameters for the simulation [3]

The test bed consists of about 20 OpenWay[1] meters manufactured by Itron and connected to each other over a RFLAN mesh network (Fig.1). They in turn are connected via a cell relay to an Ethernet switch. The Ethernet switch in turn connects the meters to the collection engine back end. The collection engine is the interface through which commands are issued to the meters. The Ethernet switch also connects to Amilyzer, the IDS.

We measured the Ping and Remote Disconnect commands on the TCIPG test bed. The Ping command sends a message back and forth to a meter, we used this to determine the communication time between a meter in the NAN and the DCU. The Remote Disconnect removes a meter from the NAN, and would be used by an attacker in our control flow attack to remove meters from the NAN once the DCU is compromised.

The test bed does not currently support encrypted traffic, so the times measured do not include encryption overhead. Additionally, we used the time for a remote disconnect command as the time to compromise a node because it involves communicating with a node, some processing, and then a response. We recognize this may have inaccuracies, but it was the best proxy available. The test bed experiments are limited due to the practical constraint that the meters are not programmable. Therefore, we were unable to inject attack traffic for example, or put timing probes for the code executing on a meter.

V. Evaluation

A. System Description

SecAMI takes three inputs. The first is the network topology of the NAN for the simulation. This consists of the nodes in the network, and their connections. SecAMI includes a tool to automatically generate a network of a user-specified size with connections between nodes generated randomly according to a power-law distribution with a parameter α of 2.5. The second and third parameters are timing information, on the time to

[1]This paper in no way suggests that there are vulnerabilities in OpenWay meters. They are just used to get realistic time measurements and we could have used meters from any other manufacturer with the same effect.

detect an attack, and the time it takes an attack to propagate from one node to a neighboring node. These times can be measured in a controlled environment, such as a test bed, and then entered into SecAMI to determine the resiliency of a large-scale network to the attack.

B. Simulator

The simulator has two parts. The first generates a network topology for the simulation, as described below. This allows us to vary the number of nodes in the network, as well as their connectivity to one another. The second part consists of a series of rules for how the attack proceeds and is detected. This allows us to examine how a hypothetical attack on the network against a given IDS model would proceed. At each time step we can observe the number of connected, compromised, and disconnected nodes (Fig. 2).

1) Network Topology: The first part generates mesh networks with a user specified number of nodes, and maximum number of connections per node. The actual number of connections a given node has a random variable with values from 1 to the specified maximum number of neighbors. Additional connections are added to avoid partitions in the network. For our evaluation, the graph generator is used to create networks of three different sizes: 50, 100, and 150 nodes. For each of these sizes, we create a subclass of graph based on the maximum number of connections per node, there are three such subclasses: 2, 6, and 10 maximum connections. We create 5 random, independent instances of each of these subclasses, and report simulation results averaged across the five instances for each of the 9 size $\times$ maximum degree combinations.

For each node in the graph, a power law random number generator is used to determine the number of connections that node should have. Then, for each node in turn, it randomly selects a node it is not yet connected to, but that still has connections available, and connects to that node. This is repeated until the current node is out of its allotted connections, and then the next node is considered. If no node exists that still has connections remaining, a new node is created to satisfy the current node's need for a connection.

2) Simulation: The second part is the simulator itself. It implements our attack and response models as described below. We use the simulator in two ways. First we use it to determine a general relationship between attack time, detection time, and network survivability. For this, we input a fixed set of attack, and detection and response times. The response in our simulations is always initiated from the DCU, under the assumption that it is more secure than individual meters. Some sample attack and response strategies are shown in the table below.

The key inputs are the NAN topology, the compromise time, the detection time, and the communication time between nodes. From those, it simulates the attack, modifying the NAN as it goes. The simulation terminates when there are no more compromised nodes, or the DCU itself has been compromised. At the end, it determines how many nodes are still reachable from the DCU (0 if the DCU itself has been compromised),

TABLE I
SOME OF SECAMI SAMPLE ATTACK AND RESPONSE STRATEGIES

Sample attack strategies	*Sample response strategies*
• Distributing malware • Getting root access on a meter • Compromising the DCU • Sending remote disconnect command from the DCU	• Disconnecting a suspect meter from the network • Sending a remote re-keying command • Rebooting a suspect meter

and reports this as a percentage of the original nodes in the NAN. Other metrics can also be collected (though not done for our evaluation), such as the throughput of data reaching the DCU, the average latency of data reaching the DCU, the consumption of bandwidth in the NAN.

The simulation contains two levels. The first level cycles through the graphs, attack to detect time ratios, and attacker starting points. The second part takes these settings and runs the actual simulation, generating a data point. During the simulation, it maintains a calendar of events (implemented as a priority queue), for instance when a node will be reached, or compromised, or when the fact that a node has been compromised will be detected. The first event to be scheduled is when the starting point of the attack will be compromised. Once that happens, a detection event is scheduled for that node, and an event for the spread to the next node is scheduled. This repeats until there are no more events remaining, which means that the attack has been stopped, or the DCU is compromised.

3) Simulation Rules: The following are the detailed rules for both the attack scenario, and our response to an ongoing attack. Once a node has been compromised, the attacker can move to any of its neighbors, and makes a random choice about which neighbor to attack next. This move takes hop time, after which the attacker has compromised the new node. This can easily be made probabilistic by assigning a pre-specified probability for the successful compromise of a vulnerable node. Back at the original node, the attacker selects another node to compromise. The attacker can attack all neighbors of a node once he has compromised it. An attack is detected after hop time times the shortest path to the DCU (computed using Dijkstra's algorithm), plus a constant detection time. Once it has been detected that a node has been compromised, that node is removed from the NAN. Further, any ongoing attack from

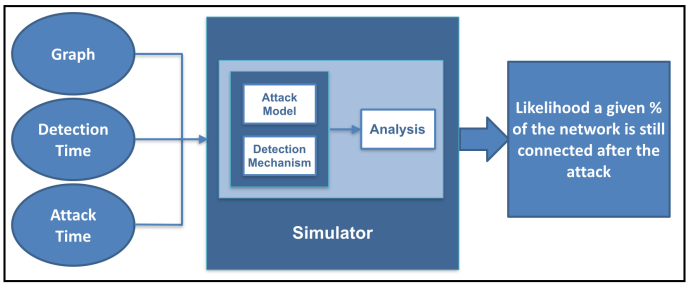

Fig. 2. SecAMI Architecture

that node is terminated, and its target is not compromised. Consequently, the DCU has to detect that some node has been compromised before it can compromise other nodes in order to stop the attack. This becomes more likely as the attack gets closer to the DCU.

We investigated two different responses to attacks using SecAMI. The first is the simplest: simply disconnect a node from the network. A subtler approach is to change the cryptographic key for the network, thus rendering the compromised meter incapable of communicating, but still able to function as a traditional "dumb" meter. Both these response strategies have been proposed and evaluated in the context of AMI [10].

VI. Results

A. General relationship

The main goal of SecAMI evaluation is to answer questions such as which attack scenarios are "under control", i.e., cause a tolerable level of damage to the network, in terms of detection and response timing and which are not. To achieve this goal, we simulate a variety of network topologies and get the average of results over all graphs, in order to avoid dependence on a particular topology, and achieve a more general result on the class of graph involved. Furthermore, we simulate the attacker starting at all possible meters in the NAN and average the results for a given topology.

We created graphs with 50, 100, and 150 nodes. For each node level, we created five graphs with 2, 6, or 10 maximum additional connections. We were primarily interested in the effect of the ratio between the time to compromise a node to the time to detect and respond to a node compromise. As such the absolute values did not matter, so we used 1 to 10 as the compromise times, and 0 to 9 as the detection times. This generated 58 unique ratios, giving us a sufficient number of data points to plot. We used every node in the graph in turn as the starting point for the attack. We generate results for each of the nine size $\times$ node degree combinations.

We first show the impact of having the attacker start close to the DCU or far from it when the response is to remove the node from the graph. There are two factors at work here: the closer to the DCU an attack is, the faster it can respond due to lower transmission time. However, an attack closer to the DCU is potentially capable of doing more damage by disconnecting more of the graph. We find that the mitigation time advantage outweighs the added vulnerability, and that the network availability is higher for attacks starting near the DCU, by an average of 18.7% (Fig. 3) starting with a attack to detect ratio of 1.4. The network's ability to reconfigure itself (basically changing the connectivity) also contributes to this, making it hard for any node, even one close to the DCU to disconnect large amounts of the graph if removed.

Next, we consider the attacker starting at every meter, and average the results together (Fig. 4). We show results for three of the nine scenarios, combined onto one graph. Namely, for 50 nodes and 2 connections, 100 nodes and 6 connections, and 150 nodes and 10 connections. These were chosen to show representative results for each node and connection category.

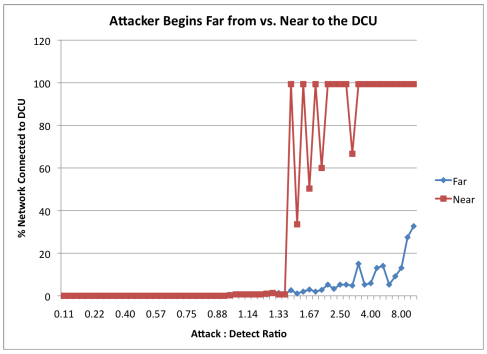

Fig. 3. Results when the attack begins far from the DCU vs. near the DCU

As the graph below shows, the ratio between the attack time and the detection time needs to be at least 3:1 to preserve a significant portion (above 90%) of the network, regardless of the size or connectedness of the topology. In other words, the time to detect and respond must be at most one third of the time to compromise a node. A sharp spike happens at this ratio and a large part of the network is available if the detection and response are faster than this.

B. Re-key Response

We also show the impact of changing the network's cryptographic key in response to an attack instead of removing the compromised node. Once the key has been changed, the compromised node can no longer communicate with the rest of the network. As a result, any ongoing attack in which that node is involved is terminated. We experimented with this technique on both highly connected networks (maximum of 10 connections per node) and sparsely connected networks (maximum of 2 connections per node) of network size 50, 100, and 150. Changing the key of the network is a more effective response than simply removing the compromised node from the network. For the sparsely connected network, the required attack to response time network for high availability after the attack drops to 1.4 (note that the attack has a greater impact on the smaller network, Fig. 5). The highly connected network fares even better, with the required ratio dropping to 1.14 (Fig. 6). This intervention is more effective because it will reach nodes that are under attack sooner than it will reach the compromised node itself thereby preventing the attack spread.

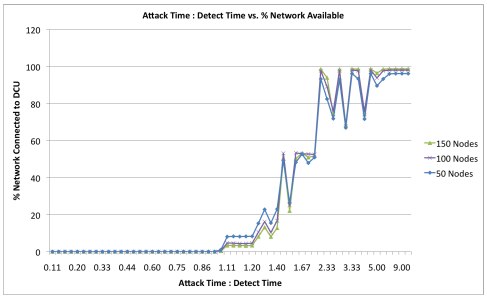

Fig. 4. Results for graphs with 50,100, and 150 nodes

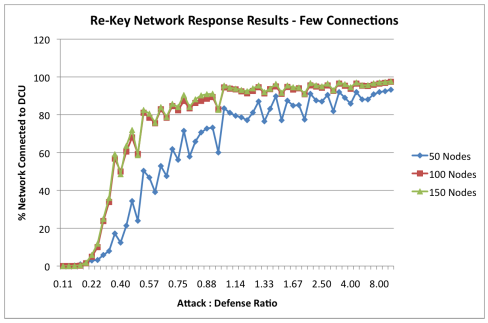

Fig. 5. Results for the Re-key network response with few connections

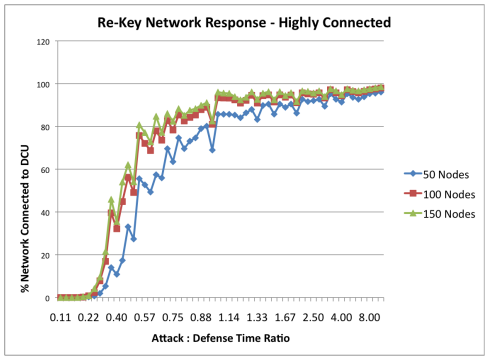

Fig. 6. Results for the Re-key network response with many connections

C. Case Study

We performed a case study of a control flow attack based on our data from the TCIPG test bed. For this case study, we used the Ping time (III.C.1) as the network transmission time between two neighboring meters. For the attack, we had a remote disconnect command being sent from the DCU. We found experimentally that a remote disconnect time on the testbed is quite slow, approximately 10 seconds. For the detection, we had the node send the remote disconnect command to the Amilyzer for its analysis and then Amilyzer's processing time before it flags the detection. Finally, for response, we had a command go out from the DCU to the compromised node. We found that the ratio of attack time to detect and respond time was very high, 94.9. Our detection time may be unrealistically fast, because we are assuming that something detectable is transmitted towards the DCU as soon as a node is compromised. Under the experimental conditions, we found that 90% - 98% of the network remained connected (Table II).

TABLE II
RESULTS FOR THE CASE STUDY WITH 20, 50, AND 100 NODES

Case Study	
Meters	% Network Alive
20	90.66%
50	96.19%
100	98.03%

VII. CONCLUSION

The paper proposes the tool SecAMI that allows a network operator to do risk assessment for Advanced Metering Infrastructure (AMI). It allows one to configure the kind of attack - the speed, the vantage point of the adversary, etc., the kind of detection, and the kind of response - the time for disconnecting a suspect node or initiating a re-keying of cryptograph keys. Under the configuration, the simulation can determine various output metrics. We present results for the percentage of the network that is connected to the Data Concentrator Unit (DCU) under attack conditions. The tool will allow an administrator to determine how much faster detection and response need to be for keeping a certain fraction of the network functional. Our experimental results show that there is a critical threshold for IDS speed. To reach this threshold, either IDSs can be sped up (if need be), or more IDSs can be deployed in a distributed manner in a given NAN. Thus, centralized IDS might not work efficiently in scalable systems in term of detection time, especially with a Distributed Denial of Service (DDoS) attack or remote disconnects on the AMI. This emphasizes the promise of distributed IDS technology for AMI as proposed in [11]. Future work should focus on evaluating several attack methodologies and detection mechanisms. Additional types of attack should be considered that may affect the high availability of any AMI. Furthermore, SecAMI will be used to find the "optimal" distribution of intrusion detection sensors in a distributed IDS over a scalable AMI.

REFERENCES

[1] R. Anderson and S. Fuloria. Who controls the off switch?. IEEE SmartGridComm-10, pages 96-101, 2010.
[2] National Electric Sector Cybersecurity Organization Resource (NESCOR). Electric sector failure scenarios and impact analyses. Technical report, EPRI, 2013.
[3] R. Berthir, W. Sanders. Monitoring Advanced Metering Infrastructures with Amilyzer. Proceedings of CESAR: The Computer and Electronics Security Applications Rendezvous, Rennes, France, Nov. 19-21, 2013.
[4] D. Grochocki, J. Huh, R. Berthier, R. Bobba, W. H. Sanders, A. Cardenas, and J. Jetcheva. AMI threats, intrusion detection requirements and deployment recommendations. In the IEEE International Conference on Smart Grid Communications (SmartGridComm), pages 395-400. IEEE, 2012.
[5] I. Shin, J. Huh, Y. Jeon, and D. Nicol. A Distributed Monitoring Architecture for AMIs: Minimizing the Number of Monitoring Nodes and Enabling Collided Packet Recovery. In Proc. of the first ACM workshop on Smart energy grid security (SEGS), 2013.
[6] W. Temple, B. Chen, and N. Tippenhauer. Delay makes a difference: Smart grid resilience under remote meter disconnect attack. In Proc. of the IEEE Conference on Smart Grid Communications, 2013.
[7] A. Almutairi, T. Shawly, A. Basalamah, A. Ghafoor. Policy-Driven High Assurance Cyber Infrastructure-Based Systems. the 15th IEEE International Symposium on High-Assurance Systems Engineering, 2014.
[8] A. Clauset, C. R. Shalizi, and M. E. J. Newman, Power-law distributions in empirical data. SIAM Rev., 51, 661-703, 2009.
[9] SecAMI tool. [Online]. Available: https://github.com/nburow/SecAMI/
[10] A. Fawaz, R. Berthier, W. H. Sanders, Cost modeling of response actions for automated response and recovery in AMI. In the IEEE International Conference on Smart Grid Communications (SmartGridComm), pages 348-353. IEEE, 2012.
[11] Y. Zhang, L. Wang, W. Sun, I. Green, M. Alam et al., Distributed intrusion detection system in a multi-layer network architecture of smart grids. IEEE Transactions on Smart Grid, vol. 2, no. 4, pp. 796-808, 2011.