Using SANs to Model DDoS Attacks in Simple AMI Networks

Zach Yordy
Dept. of Electrical and Computer Engineering
University of Illinois, Champaign, IL 61820

Huaiyu Zhu
Dept. of Electrical and Computer Engineering
University of Illinois, Champaign, IL 61820

I. Introduction

The advanced metering infrastructure (AMI) system uses metering devices to gather and analyze energy usage information. The emergence of the use of AMI is an important step towards building an efficient and secure smart grid. Various communication models, protocols, and devices can be combined to form the communication backbone of an AMI network. In North America, the major deployment of AMI is based on a Radio Frequency Mesh network architecture, wireless metering devices, and the ANSI C12.22 application-level protocol suite. However, due to its nature of publicly accessible wireless communication, AMI is vulnerable to various cyber-attacks [1]. In this project, we model an exploit of the C12.22 protocol that causes DDoS-like behavior in AMI networks by using a stochastic activity network (SAN) to get some insights on how such an attack might overwhelm the network. Earlier studies [2, 3] have shown in simulation that these attacks can be highly effective, however, an analytical model is also necessary to help us further understand important characteristics of this attack.

II. Problem Description

In this project, we aim to model a DDoS attack which exploits a vulnerability in the *trace service* of the C12.22 protocol that might be observed in the metering network. The C12.22 protocol, widely used for AMI systems, defines how to exchange information between AMI devices. It allows for a various number of services to be run, including read and write services, among many others. Specifically, it provides a trace service that is used to find out the route that a particular C12.22 message traverses. While the trace service is a convenient tool with which network administrators can debug the network, the design does not include any security features, which makes it easily leveraged by malicious users to launch DDoS attacks.

Under normal use, the trace service performs its intended function admirably. The node requesting the trace service is provided with a list of IDs of the relays through which the message was passed on its way to the destination. This information is obtained by forwarding the message through the network. At each hop, the node appends its own node ID, called an *ApTitle*, to the message. The final message is then sent back to the requestor through the same route. Suppose a C12.22 node with unique *ApTitle* `X` wants to trace the route to another node whose *ApTitle* is `Y`. The requesting node initiates a trace service request, including the *ApTitles* `X` (the source, or *Calling Aptitle*) and `Y` (the destination, or *Called ApTitle*), and sends the message out through the network. When a C12.22 relay responsible for forwarding the message receives the trace service request message, it extracts the *Called ApTitle* to find the next hop through which to route the message. Then, it adds its own *ApTitle* to the message payload. Once destination `Y` receives this trace service request, it initiates a *trace service reply* with *Calling ApTitle* `Y` and *Called ApTitle* `X`, containing the route obtained using the service request. This reply message is finally delivered to the original requestor without any changes during transmission.

Note that the message size of the trace service request is growing along the transmission path (due to an *ApTitle* being appended by each relay). In this way, the service is vulnerable to attack. Figure 1 shows this amplification DDoS attack scenario in an AMI network. The figure shows two

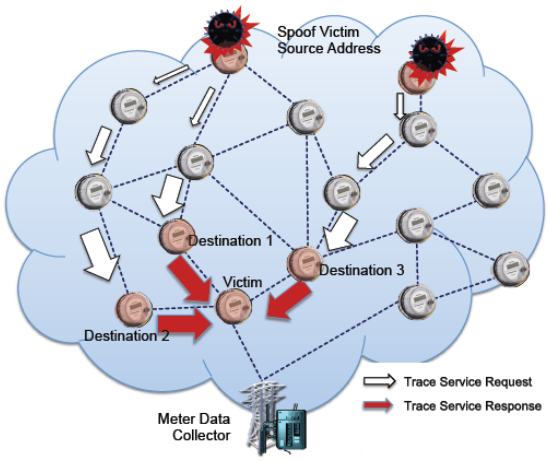

Fig. 1. DDoS scenario in AMI network.

compromised meters (C12.22 nodes) controlled by malicious users. The compromised meters use a large portion of their bandwidth to send out trace service requests to a set of carefully selected destinations. However, the sending meters spoof their *ApTitles* so that any replies will be aimed at a victim meter. The request messages can grow fairly large due to multiple hops in the network (indicated by the increasing size

of the white arrows in the figure). Once the destination meters receive the requests, they will send a reply message to the victim meter because of the spoofed addresses encapsulated in the requests. As illustrated by Figure 1, a large number of long reply messages are simultaneously sent to the victim. This traffic, now lengthened by the attack, might overwhelm the victims bandwidth or crash a running application on the victim node with a buffer overflow.

In an RF mesh network, the attacking traffic can jam quite a large area surrounding the victim. Since the victim is not able to respond to legitimate requests, a successful DDoS attack can be performed. The attacker would most likely choose an important victim node such as an egress node to the network. During such a DDoS attack, most legitimate traffic would be dropped before it could even get back to the utility.

III. Model Description

By using simulation, we can easily set up an arbitrary network topology and define communication protocols to analyze the effectiveness of the attack. However, the simulation results are largely dependent on the simulated topology of the network as well as details about the capability or bandwidth of the devices. Thus, a simple analytical model might be a better solution to reveal the true nature of the effects of such an attack, although assumptions about capacity, bandwidth, and topology still have to be made. A typical DDoS attack has the following features:

- Some of the nodes are compromised, but the majority of the network is not.
- The attacking nodes send out packets much more frequently than the legitimate nodes do.
- The high volume of traffic jams the bottleneck of the network (one or several nodes).

To test the sensitivity of the attack with different network topologies, we modeled the network with a both mesh topology and a tree topology as shown in Figures 2 and 3, respectively. The dynamic routing protocol was modeled by assigning a fixed probability to each outgoing link. In our model, all outgoing links have equal probability. To minimize the state space, we constructed a simple subnetwork with six nodes and one or two malicious nodes. We also modeled the network without any attackers. We tried to use fairly common metrics for AMI networks in our models. We assumed an average message size of 100 bytes, and a bandwidth of 250 bps. We also assumed that background traffic accounted for approximately 10% of the total bandwidth. This gives us a rate of

$$.1 \times \frac{250 \text{ bytes}}{1 \text{ second}} \times \frac{1 \text{ message}}{100 \text{ bytes}} = 0.25 \text{ messages per second, or}$$

$$\frac{.25 \text{ messages}}{1 \text{ second}} \times \frac{3600 \text{ seconds}}{1 \text{ hour}} = 900 \text{ messages per hour.}$$

Therefore, background traffic was modeled by a Poisson process with rate 900 messages/hour. The transmission of each message was modeled by a Poisson process with rate 9000 messages/hour, representing the available bandwidth. Finally, the attackers were modeled as sending out packets at a much higher rate than legitimate nodes. Messages were generated by attackers at a rate of 90,000 per hour on average, but remember, the attacker could only use all of its available bandwidth to send messages. A generation rate of 90,000 messages per hour would only guarantee that there were usually messages ready to be sent. A complete description for these models can be found in Appendix A, but the SANs for the mesh and tree networks are shown in Figures 4 and 5, respectively.

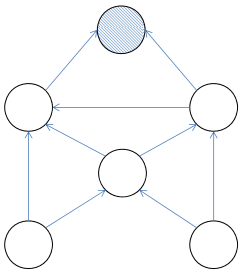

Fig. 2. Modeled mesh network.

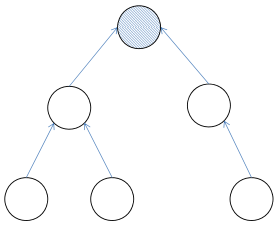

Fig. 3. Modeled tree network.

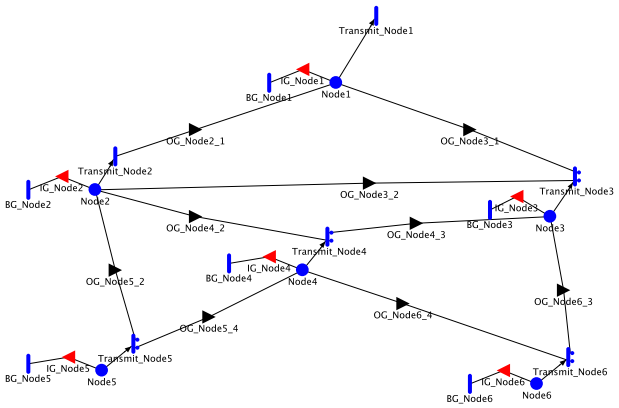

Fig. 4. Mesh network SAN model.

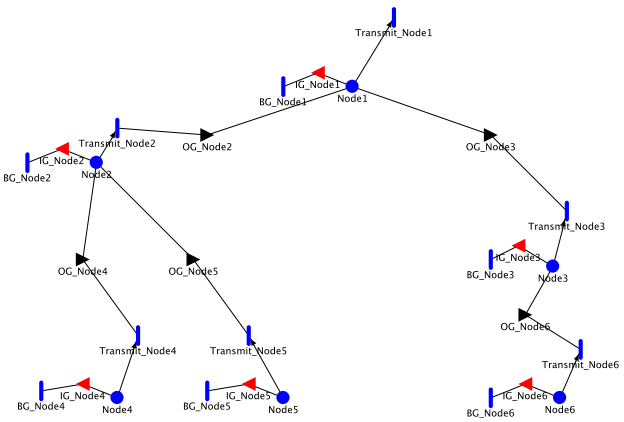

Fig. 5. Tree network SAN model.

IV. Results

The full outputs of the SAN modeling are attached in Appendix B. With these models, we saw the following results:

Percentage of Time Spent Busy or With a Full Queue

	Queue size of 3, No attackers	Queue size of 3, 1 attacker	Queue size of 3, 2 attackers
Mesh Network, Top Node Busy (Messages in Queue > 0)	**53.16%**	**79.66%**	**85.64%**
Mesh Network, Top Node Full (Messages in Queue = Max)	**9.27%**	**29.09%**	**37.35%**
Tree Network, Top Node Busy (Messages in Queue > 0)	**53.65%**	**81.86%**	**91.27%**
Tree Network, Top Node Full (Messages in Queue = Max)	**9.63%**	**32.03%**	**47.70%**

	Queue size of 5, No attackers	Queue size of 5, 1 attacker	Queue size of 5, 2 attackers
Mesh Network, Top Node Busy (Messages in Queue > 0)	**57.82%**	**90.81%**	**95.13%**
Mesh Network, Top Node Full (Messages in Queue = Max)	**17.53%**	**64.98%**	**76.85%**
Tree Network, Top Node Busy (Messages in Queue > 0)	**57.92%**	**92.01%**	**98.06%**
Tree Network, Top Node Full (Messages in Queue = Max)	**17.66%**	**68.00%**	**87.29%**

*Note: Full indicates that the node is dropping all incoming packets (DOS). All results come from a 15-minute simulation.

Fig. 6.

Let us first look at the results of modeling nodes that can buffer up to three messages at a time. With no attackers, in both the mesh network and the tree network, the egress node was busy roughly half the time, and had a completely full queue about 10% of the time. When one attacker was introduced into the network, these numbers changed quite a bit. There was a fairly sizable increase in the amount of time that the egress node was busy, and a very large increase in the fraction of time spent with a completely full queue (in both cases over triple what the values were without any attackers).

Adding a second attacker did not increase the numbers by quite as much, but still made a large impact. In the worst case, the egress node was busy over 90% of the time, with more than half of that time spent with a completely full queue, rejecting any incoming messages. In all cases, the tree network performed less admirably than the mesh network. With no attackers, the egress node of the tree network spent only a slightly larger fraction of the time busy and with a full queue, but these disparities worsened when attackers were introduced into the network. In the worst case, with two attackers, the egress node spent about 6.5% more time busy in the tree network than in the mesh network, and a whopping 27.8% more time with a completely full queue. This shows that although the tree and mesh type networks perform similarly under normal circumstances, an abundance of messages in the network congest the top node much more quickly.

In order to start small and grow our network, we decided to start with a small number of nodes that were only able to buffer three messages at a time, the results of which have just been discussed. Our state space was 4096 states, accounting for 0, 1, 2, or 3 messages in each of the six nodes' queues. A more realistic node has a queue that can buffer five or even ten messages at a time. With a queue size of five messages, the state space quickly grew to 46,656 states. With a queue size of ten messages, the state space would have been (a very unmanageable) 1,771,561 states. We decided to stick with a buffer size of five states as the maximum to ensure that our state space was not too big to solve. Similarly, six nodes can model some very diverse networks, but it does not take long to explode the state space into unmanageable territory by only adding a few nodes. Since differences between types of networks were still distinguishable with as little as six nodes, we decided it would be unnecessary to complicate the problem with little gain by adding a small number of additional nodes.

With the more reasonable queue size of five, the egress node spent more time with messages in its queue across the board. Since each node could hold more messages, fewer messages were dropped by each node, delivering more messages that had to be processed by the egress node. All broad conclusions gained from the first set of data also apply to this set, although there is one major difference that can be pointed out. With a queue size of five, adding just one attacker much more quickly overwhelms the network. With a queue size of three, the amount of time spent busy increases by about 50% when the first attacker joins the network. Similarly, the amount of time spent with a full queue increases roughly by a factor of three. With a queue size of five, the egress node saw an increase of roughly 57% in busy time with the addition of an attacker to the network, while time spent with a completely full queue saw an increase of almost a factor of four! The addition of the second attacker to the network did not make quite the same impact, since the network was already primarily overwhelmed at this point. In both network models, the addition of just one attacker meant that the egress node would drop packets at least 65% of the time.

V. Conclusion and Future Work

In this project, we demonstrate a DDoS attack scenario in an AMI network. We believe that understanding the behavior of the attack is essential to building a secure AMI system. The targeted network is constructed with a simple but representative topology. In addition, we model legitimate and malicious behavior using a stochastic activity network (SAN). By solving the models we built, we can see many interesting results about the DDoS attack under various conditions.

In the near future, we will extend this study to investigate more sophisticated DDoS strategies. We will apply the SAN model to a larger network with a more realistic configuration which can be obtained by collecting data either from a real traffic trace or generated by a simulator. We can also compare the results to simulation results to verify its accuracy. If we can fully comprehend DDoS attacks in AMI, we will be able to design an efficient defense mechanism to limit DDoS. This defense mechanism can also be analyzed by a properly designed SAN model.

REFERENCES

[1] S. Rana, H. Zhu, C. W. Lee, D. M. Nicol and I. Shin, *The Not-So-Smart Grid: Preliminary Work on Identifying Vulnerabilities In ANSI C12.22*, GC'12 Workshop, 2012.

[2] D. Jin, C. Lee, D. Nicol, I. Shin, and H. Zhu, *Simulation-based Study of Distributed Denial-of-Service Attacks in Advanced Metering Infrastructure*, INFORMS Annual Meeting, Charlotte, NC, USA, November 2011.

[3] D. Jin, Y. Zheng, H. Zhu, D. M. Nicol, and L. Winterrowd, *Virtual Time Integration of Emulation and Parallel Simulation*, Proceedings of the 26th Conference on Principles of Advanced and Distributed Simulation (PADS), Zhangjiajie, China, 2012.