LA-UR- 10-08350

Approved for public release; distribution is unlimited.

Title:	Evaluating Resilience of DNP3-Controlled SCADA Systems against Event Buffer Flooding
Author(s):	Dong Jin, David M. Nicol, Guanhua Yan
Intended for:	The 41st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'11)

Form 836 (7/06)

Evaluating Resilience of DNP3-Controlled SCADA Systems against Event Buffer Flooding

Dong Jin, David M. Nicol
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
{*dongjin2, dmnicol*}*@illinois.edu*

Guanhua Yan
Information Sciences (CCS-3)
Los Alamos National Laboratory
Los Alamos, NM 87545, USA
ghyan@lanl.gov

***Abstract*—The DNP3 protocol is widely used in SCADA systems (particularly electrical power) as a means of communicating observed sensor state information back to a control center. Typical architectures using DNP3 have a two level hierarchy, where a specialized data aggregator device receives observed state from devices within a local region, and the control center collects the aggregated state from the data aggregator. The DNP3 communication between control center and data aggregator is asynchronous with the DNP3 communication between data aggregator and relays; this leads to the possibility of completely filling a data aggregator's buffer of pending events, when a relay is compromised or spoofed and sends overly many (false) events to the data aggregator. This paper investigates how a real-world SCADA device responds to event buffer flooding. A Discrete-Time Markov Chain (DTMC) model is developed for understanding this. The DTMC model is validated by a Möbius simulation model and data collected on real SCADA testbed.**

I. Introduction

Supervisory control and data acquisition (SCADA) systems are used to control and monitor critical infrastructure processes including electrical power, water and gas systems. As such, SCADA systems are critical to our daily lives. The United States is currently conducting a major upgrade of its electrical system, making the grid "smarter", but in doing so adding more vulnerabilities. We have seen the consequence when large areas lose power for an extended period of time[1][2][3]; the obvious threat is that attackers harm the grid infrastructure through largely electronic means.

The Distributed Network Protocol v3.0 (DNP3) is the most widely used SCADA network communication protocol in North America (approximately 75%) [4]. Designed to provide interoperability and as an open standard to device manufactures, DNP3 has no notion of security, and most DNP3 devices lack identity authentication, data encryption and access control. Although some enhanced versions of DNP3, such as DNP3 Secure Authentication [5] or DNPSec [6], have been developed but yet still under evaluation phase, the majority of DNP3-controlled devices in SCADA networks are currently working with little protection.

Most of the existing works on DNP3 security scrutinize potential security risks inherent in the DNP3 protocol specifications. A taxonomy of attacks across all layers of the DNP3 protocol has been summarized by East *et al.* to show how vulnerable the protocol is [7]. In this work, we analyze how DNP3-controlled systems respond to event buffer flooding. The adversary can simply send many data events to a device that temporarily buffers SCADA data before they are retrieved by a control station. The event buffer is filled so as to prohibit the buffering of critical alerts from legitimate devices, negatively impacting the control station's situational awareness.

In a nutshell, the main contributions of this paper are: (1) We demonstrate how a real-life DNP3 device responds to event buffer flooding in a laboratory setting at TCIP, located in the University of Illinois at Urbana and Champaign[1]; (2) we develop a DTMC model studying the effectiveness as a function of various behavioral parameters. The analytical model has been validated by the data from real testbed as well as a simulation model created in Möbius [8]; (3) we suggest some countermeasures based on our analysis.

The remainder of the paper is organized as follows: Section II gives an overview of DNP3-controlled SCADA networks. Section III describes the threat model. Section IV introduces a potential vulrability in DNP3 slave devices. Section V explores how a real data aggregator responds to event buffer flooding. Section VI presents a DTMC model and a simulation model to evaluate the effect of event buffer flooding, and compares the two models with results from real data aggregator. Section VII discusses countermeasures and Section VIII describes related work. Finally, we draw concluding remarks in Section IX.

II. DNP3 Overview

The DNP3 protocol carries control and data communication among SCADA system components. It is a master-slave based protocol, where a master issues control commands

[1]http://www.iti.illinois.edu/content/tcip-trustworthy-cyber-infrastructure-power-grid

to a slave and a slave collects data that is returned to the master. Typically a utility has a central control station for managing and monitoring its portion of the grid. The control station acts as a top-level DNP3 master, gathering data from substations, displaying the data in a human-readable formation, and making control decisions. A *data aggregator* located in a remote substation serves both as a DNP3 master to control and collect data from monitoring devices, **and** serves as a DNP3 slave to transmit (on demand) all of the data it has collected back to the control station. Figure 1 depicts the typical two-level architecture. DNP3 devices were widely used on serial links in old days, and many of them are still in use. Newer DNP3-controlled networks use TCP/IP-based connections where the DNP3 message is embedded as a payload of the underlying layer's packet. As a result, DNP3 can take advantage of Internet technology and to conduct economical data collection and control between widely separated devices. Our work focuses only on the DNP3 over TCP communication.

The data collected at the DNP3 slave is classified as being one of *binary data, analog data* or *counter data*. Binary data are used to monitor two-state devices, e.g. a circuit breaker is closed or tripped; analog data carry information like voltage and current on a power line. Counters are useful for reporting incremental values such as electricity usage in kilowatt hours. Data are transmitted to a master via two modes: polling and unsolicited response. In polling mode, a master periodically asks all the connected slaves for data, typically in a round robin fashion. Polling mode can be further divided into integrity polling and event polling. An integrity poll simply collects all static data with their present values. A event poll only collects DNP3 *events* that flag important changes, e.g. when a binary data changes from an on to an off state or when an analog value changes by more than its configured threshold. In unsolicited response mode, a slave spontaneously sends DNP3 events to its master. A DNP3 master usually issues an integrity poll at start-up and then primarily uses event polling, with periodic refreshes with an integrity poll. The period of integrity polling (e.g. hourly) is generally much longer than the period of event polling (e.g. a few seconds).

A DNP3 slave that is configured to use *unsolicited response mode* may deliver data to a DNP3 master without being polled. This is useful for reporting state changes where a reaction is time-critical.

III. Threat Model

Buffer flooding attacks have been commonly observed in many types of communication networks, such as the Internet. In this work, we study how DNP3-controlled SCADA systems respond to buffer flooding attacks. The buffer flooding attacks assume the ability to access the substation network through some entry points, such as the utility's enterprise network or even the Internet. Although the flooding targets

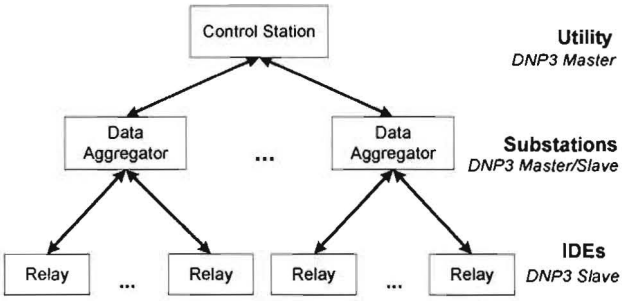

Figure 1. Two-level architecture of a DNP3-controlled SCADA network

are the data aggregators with a substation, the attacks do not assume the ability to compromise a data aggregator. In order to flood the data aggregator's event buffer, the adversaries must establish a connection with the data aggregator as a legitimated relay, which can be achieved by either spoofing a normal relay or compromising a victim relay.

No authentication is currently supported in DNP3 protocol to prevent the adversaries from spoofing the relays. The adversaries can suppress a normal relay by redirecting the victim relay's traffic to itself with techniques such as ARP spoofing and then spoof the victim relay to re-establishing a new connection with the data aggregator. The adversaries can also act as a secret middle man between the victim relay and the data aggregator and aggressively replay unsolicited response events captured from the victim relay to exhaust the buffer resource.

The buffer flooding attack can also be launched from compromised relays. The reality is that the security of many commercial relays is only provided by having each relay require a password. Once the single password is captured, the relay is fully compromised. Unfortunately, bad password practices have always been observed in substation-level networks. Many operators do not change the default password for the sake of convenience. The magic words "otter tail" was definitely listed at the top of an adversary's dictionary, because it was used by a major relay manufacturer as a default password and surprisingly was observed to remain unchanged over many SCADA systems. Furthermore, most relays do not have a limit on the number of log in attempts, which could easily make a typical automated password cracker software effective.

IV. The Vulnerability

A data aggregator serves as a DNP3 master to relays and as a DNP3 slave to the control station; one can think of it as having a master module and a slave module. The master module queries relays and stores received events into the slave module event memory. The data aggregator responds to queries from the control station by reading out portions of its slave module event memory. The vulnerability arises because the aggregator's polling of relays is performed

asynchronously with the control station's queries to it. The slave memory is therefore a buffer, filled by responses from relays and emptied by a control station query.

Two types of event buffers are commonly used in commercial DNP3 slave devices: *sequence of event* and *most recent event*. The former simply stores all received data in the event buffer. Every new event occupies new buffer space; if the buffer is full then the event is discarded. This type of buffer is useful for various applications including grid state estimation and trend analysis. By contrast, a most recent event buffer reserves space for each individual data point that the aggregator might acquire. When an event arrives, all the buffer locations associated with data points it carries are overwritten, regardless of whether their current values have first been read out by a control station query.

The potential vulnerability of interest arises with sequence of event buffers, because it is feed by all slaves from which the data aggregator acquires data. The attack has a compromised DNP3 slave (or an adversary on the network successfully pretending to be a DNP3 slave) send so many unsolicited events that the buffer is filled, and events from uncompromised slaves are lost until the buffer is emptied by a query from the control station.

V. Experiments on Data Aggregator

A. Buffering Mechanism Experiments

The DNP3 specification describes the general guidelines on event buffer semantics and leaves the implementation to vendors [9]. The vendor's implementation is generally not publically available. Therefore, in order to study how DNP3-controlled systems respond to event buffer flooding, we need to first conduct experiments on a real data aggregator to understand its buffering mechanism.

The test data aggregator supports the three data types mentionned before (binary, analog, and counters). Each data type has an independent buffer. To understand how each buffer works, we connected the device with relay A and relay B as two DNP3 slaves, and configured one host as a DNP3 master that plays the role of a control station. Initially, we set the size of every buffer to 5, and cleared all the buffers in the data aggregator by issuing sufficient integrity polls from the control station. Let A_i and $B_i (i = 1, 2, ...)$ be the unsolicited response event sent from relay A and relay B to the data aggregator respectively. Each event contains the same one data point with a different value. Figure 2 is the time sequence diagram showing the experimental results for all three data types.

The experimental results indicate that

- buffers of all three data types have the first come first serve (FCFS) scheduling mechanism.
- the counter event and binary event buffers use "sequence of event" mode, and thus are vulnerable to buffer flooding. Once the buffer was full, any incoming events were dropped, and the event buffer overflow indicator bit in the head of DNP3 message was observed to be set to true.
- the analog event buffer uses "most recent event" mode; once the same data point was received more than once before being read out, its storage location was overwritten. Analog event buffers are immune to flooding, because an adversary's flooding affects only the buffer space allocated for the adversary's device.

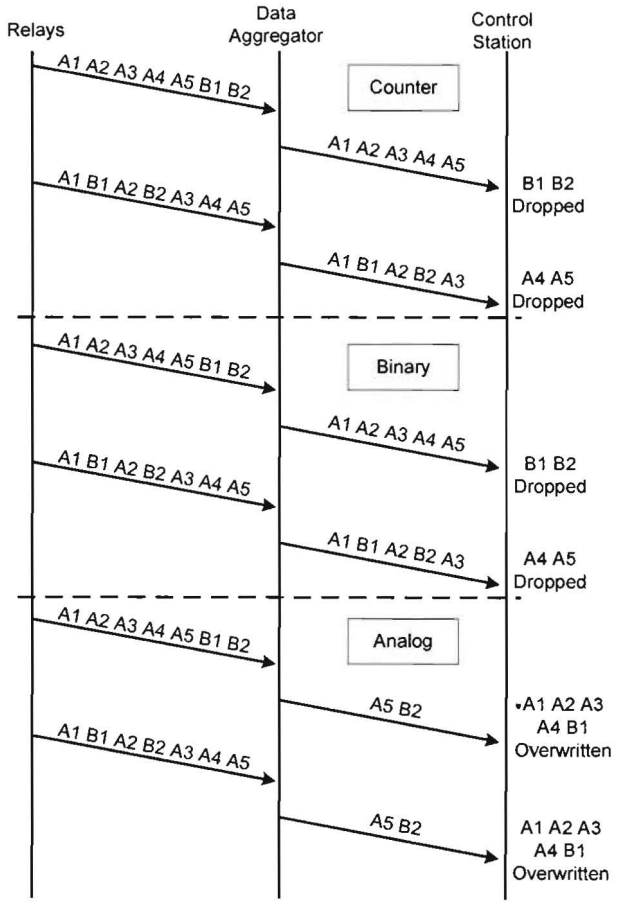

Figure 2. Time Sequence Diagram: Experimental Results for Revealing Data Aggregator's Buffering Mechanism, Buffer Size = 5

B. Buffer Flooding Experiments

The next experiment studies how the data aggregator responds to buffer flooding. The data aggregator serves as DNP3 master to two relays, and as a DNP3 slave to a control station. The data aggregator polls the relays every 10 seconds. In addition the relays also send unsolicited response events to the data aggregator. Assume one relay is captured or spoofed by the adversary and it can generate many unsolicited response events and stop responding to polling requests. The unsolicited response event traffic from

the adversary's relay is injected with a constant inter-event time (which we will also refer to as "constant bit rate"). A normal relay always provides 3 events in response to a polling request, and also injects unsolicited response event traffic with an exponentially distributed inter-event time, with rate parameter 3 events per 10 seconds. All the traffic contains only counter events. Each event takes a value from an sequence number (continually incremented) to facilitate us identifying which events are lost (by looking for gaps in the reported sequence numbers). For these experiments we left the counter buffer at its default size of 50 events. The control station periodically polls the data aggregator every 10 seconds.

The attacker sending rate is chosen from 1 event/sec to 20 event/sec; each experiment generates 100,000 attack events. Figure 3 shows the fraction of dropped events for the normal relay's polling and unsolicited response events, under various attacker sending rates. Both types of events start to be lost when the attack rate is 5 event/s, because the buffer fills within one polling interval. The drop fraction increases as the attacker sending rate increases, and is nearly 80% at an attack rate of 20 event/sec. The sending rate can be no larger than network bandwidth / packet size. For example, with a 10 Mb/s Ethernet connection and 100-byte packet (which contains four DNP3 counter events), an adversary might send up to 50,000 counter events per second. From this we see that the buffer can be flooded and cause significant loss of real events under attacks whose rates are far smaller than the network line rate. Of course, the control station will realize that events have been lost (because of a status bit in the DNP3 response), and a burst of unusual unsolicated events could easily be noticed if a sniffer was watching traffic (which is actually very unusual in real DNP3 contexts). The flooding attack would be most effective if launched in coordination with other attacks (perhaps even physical attacks), denying the control station's situational awareness of the state of the substation.

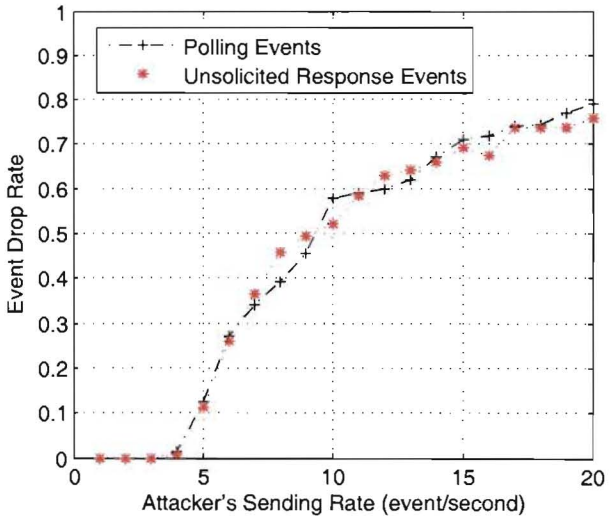

Figure 3. Fraction of Dropped Events from Normal Relay on Real Testbed

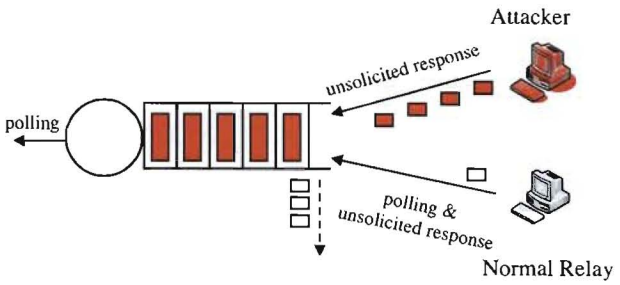

Figure 5. Queueing Diagram of the Data Aggregator's Event Buffer

VI. Modeling and Analysis

A. Analytical Model

We developed a DTMC analytical model for investigating event buffer flooding. The time-step is the control station polling interval length. The DTMC state is the buffer size at the instant a control station poll request arrives. Figure 5 depicts the data aggregator's event buffer as a queueing system. The system has three inputs: the unsolicited response events from the attacker relay, polling events and unsolicited response events from the normal relay. The shared buffer with finite size will drop any incoming events once it gets full. The output is triggered by control station's periodic polling request. Figure 4 illustrates event arrivals within a control station's polling interval. Here we assume that the control station and the data aggregator are configured to have the same polling interval.

The parameters of the analytical model are summarized as follows:

b	event buffer size
m	max #events transmitting to control station from data aggregator per control station poll
δ	control station's constant polling interval
r	adversary's unsolicited response event sending rate, events arrive in constant bit rate
λ	mean arrival rate of unsolicited response events from normal relay, event arrival follows a poisson distribution
w	number of events collected from normal relay per data aggregator's polling
S	normalized time within time-step at which bulk arrivals from normal relay poll arrive
k	time slot index, the time is slotted by the control station's polling interval
$Q(k)$	#events in the buffer at the beginning of k^{th} time slot
$A(k)$	#total arriving events during k^{th} time slot
$N(k)$	#unsolicited response events from normal relay

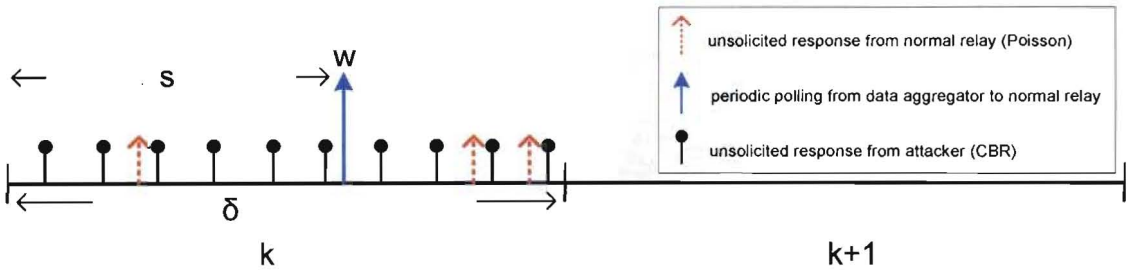

Figure 4. Timing Diagram of Event Arrivals

during k^{th} time slot

$D(k)$ #departing events polled by the control station at the end of k^{th} time slot

The queueing system can be described by

$$Q(k+1) = [min(Q(k) + A(k), b) - D(k)]^+ \quad (1)$$

The system can therefore be modeled as a DTMC, in which the time is discretized by the control station's polling interval. Let $Q(k)$ be the state of the markov chain, $Q(k) \in 0, 1, 2...b - m$. The state transition probability is derived by

$$P(Q(k+1) = j | Q(k) = i) = \begin{cases} P(i + A(k) \leq m) & \text{if } j = 0 \\ Pr(i + A(k) \geq b) & \text{if } j = b - m \\ Pr(i + A(k) - m = j) & \text{otherwise} \end{cases} \quad (2)$$

$$P(A(k) = r\delta + w + N(k)) = P(N(k) = n) = \frac{(\lambda\delta)^n e^{-\lambda\delta}}{n!}, \text{ where } n \in 0, 1, 2... \quad (3)$$

The DTMC is time-homogeneous. Let $\Pi = (\pi_0, \pi_1, ..., \pi_{b-m})$ denote the state occupancy probability vector in steady state, where π_i is probability that the DTMC is in state i in steady state.

$$\begin{cases} \sum_{i=0}^{b-m} \pi_i = 1 \\ \Pi = \Pi P \end{cases} \quad (4)$$

Let L_i be the total number of dropped events per time slot in state i, i.e. there are i events in the buffer at the beginning of the time slot.

$$L_i = ((A - (b - i))^+ \quad (5)$$

where the distribution of A is specified in Equation (3), and the dependence on k is removed from the notation as we are interested in the asympotitic behavior.

The average number of dropped events per time slot is computed as

$$E(L) = \sum_{i=0}^{b} \pi_i E[(A - (b - i))^+] \quad (6)$$

The ratio of expected dropped events of all types to expected events in a time slot is

$$\rho = \frac{E(L)}{E(A)} = \frac{E(L)}{r\delta + \lambda\delta + w} \quad (7)$$

a value which by Jensen's Inequality [?] is a lower bound on the expected fraction of all events that are dropped.

ρ bounds the overall fraction of dropped events (including attacker events); of more interest is the fraction of events dropped events from the normal relay. Define T_f to be the time required (from beginning of a time slot) for the buffer to fill in a time slot.

$$P_i(T_f = t | S = s) = \begin{cases} P(N_f = b - i - rt) & \text{if } 0 \leq t < s \\ \sum_{j=0}^{w} P(N_f = b - i - \lfloor rs \rfloor - j) & \text{if } t = s \\ P(N_f = b - i - rt - w) & \text{if } s < t \leq \delta \end{cases}$$

$$= \begin{cases} \frac{(\lambda t)^{b-rt-i} e^{-\lambda t}}{(b-rt-i)!} & \text{if } 0 \leq t < s \\ \sum_{j=0}^{w} \frac{(\lambda s)^{b-\lfloor rs \rfloor - j - i} e^{-\lambda s}}{(b-\lfloor rs \rfloor - i - j)!} & \text{if } t = s \\ \frac{(\lambda t)^{b-rt-w-i} e^{-\lambda t}}{(b-rt-w-i)!} & \text{if } s < t \leq \delta \end{cases} \quad (8)$$

where N_f is the random number of unsolicited response events from normal relay within T_f time; these events are not dropped. Time $t \in \{\frac{b-i-z}{r}, \text{ where } z = 0, 1, 2...\} \cup \{s\}$ and $0 \leq t \leq \delta$.

The average number of dropped unsolicited response events and polling events from normal relay given T_f can be computed respectively as

$$E(L_i^{ur} | T_f = t, S = s) = E(L_i^{ur} | T_f = t) = (\delta - t)\lambda \quad (9)$$

$$E(L_i^{poll} | T_f = t, S = s) =$$

$$\begin{cases} w & \text{if } 0 \le t < s \\ \sum_{j=0}^{w}(w-j)P(N_f = b-i-\lfloor rs \rfloor - j) & \text{if } t = s \\ 0 & \text{if } s < t \le \delta \end{cases} \tag{10}$$

The average number of dropped unsolicited response events and polling events from normal relay within a time slot can be derived respectively:

$$E(L^{ur}) = \sum_{i=0}^{b-m} \pi_i \int_{s=0}^{\delta} f(s) \sum_{all\ t} \bar{P}_i(T_f = t|S = s) E(L_i^{ur}|T_f = t, S = s) ds \tag{11}$$

$$E(L^{poll}) = \sum_{i=0}^{b-m} \pi_i \int_{s=0}^{\delta} f(s) \sum_{all\ t} \bar{P}_i(T_f = t|S = s) E(L_i^{poll}|T_f = t, S = s) ds \tag{12}$$

where $P_i(T_f = t|S = s)$ is normalized by

$$\bar{P}_i(T_f = t|S = s) = \frac{P_i(T_f = t|S = s)}{\sum_{all\ t} P_i(T_f = t|S = s)} \tag{13}$$

Thus, a lower bound on the expected fraction of lost normal unsolicited response events is

$$\rho^{ur} = \frac{E(L^{ur})}{\lambda \delta} \tag{14}$$

while the exact expected fraction of lost normal polling events is

$$\rho^{poll} = \frac{E(L^{poll})}{w} \tag{15}$$

ρ^{poll} is exact because w is constant in this model.

B. Simulation Model

We also built a stochastic activity network (SAN) [10] simulation model with respect to the real testbed setup in Möbius v2.3.1. Möbius was first introduced in [11], with the goal of providing a flexible, extensible, and efficient framework for implementing algorithms to model and solve discrete-event systems. SAN, which is a stochastic extension to Petri net [12], is a high-level modeling formalism supported in Möbius. SANs consist of four primitive objects: places, activities, input gates, and output gates. Activities (thick vertical lines graphically) represent actions of the modeled system that take some specified amount of time to complete. Places (circles graphically) represent the state of the modeled system. Input gates (triangles graphically) are used to control the enabling of activities, and output gates (triangle with its flat side connected to an activity) are used to change the state of the system when an activity completes. In Möbius, we can also define reward variables that measure information about the modeled system.

Figure 6 shows the core design of the event buffer flooding model. The place "EventBuffer" models the shared finite event buffer in a data aggregator. The event buffer queues events from three data sources, which are modeled as three activities: attacker relay's constant bit rate traffic, normal relay's poisson arrival traffic and normal relay's constant polling traffic, of which two are deterministic process and one is exponential process. The places "UR_Drop" and "Polling_Drop" are used to keep track of the number of dropped unsolicited response events and polling events from normal relay respectively. The fraction of dropped events are, for both types, set to be steady state reward variables for simulation study.

C. Model Validation

Both real testbed data and the simulation model are used to validate the analytical model. All the parameters of the analytical model and the simulation model are taken from the real testbed: $b = 50, m = 50, \lambda = 0.3$ event/second, $w = 3$ event/second, $\delta = 10$ seconds. Recall that S is the fraction of time between successive control station polls that elapses before the data aggregator poll delivers a bulk arrival to the buffer. We empirically determined the probability distribution of S from testbed data based on 10,000 samples and plot the empirical CDF of S in Figure 7. It is clear that S can be modeled as a uniform distributed random variable between 0 to 10. With all the parameters in analytical model and simulation model aligned well with real testbed setup, we vary the attacker sending rate from 1 event/second to 20 event/second with 1 event/second increment, and statistically compute the mean fraction of dropped events for both unsolicited response events and polling events from the normal relay. For all the reward variables in the Möbius model, the confidence level is set to 0.99 and relative confidence is set to 0.1, which means that results will not be satisfied until the confidence interval is within 10% of the mean estimate 99% of the time. For every experiment of the Möbius model, we conducted 10 independent runs with a different random seed. For each experiment, the minimum number of runs is 10,000 and maximum number of runs is 100,000. During all the experiments, the reward variables in the Möbius model are able to converge within the maximum number of runs. The degree of closeness of two sets of data are measured by the relative error. The relative error is defined as $\frac{|\hat{y}-y|}{y}$, where y is the baseline data and $\hat{y}$ are the data points to compare with the baseline data.

Figure 8 plots our estimates of the fraction of dropped events. The real data curve plots empirically observed fractions, the simulation model curve plots statistical estimates of the true observed fractions, and the analytic model plots the analytic upper bound on the true observed fractions. For the Möbius model, the results from the 10 independent runs have little variance and are extremely close to the testbed observations. The relative errors are also listed in Table I. It can be seen that the analytic estimates for both unsolicited response and polling events match those of the simulation model with very small relative error. The analytical model and simulation model also match well with the real testbed data. Therefore, the analytical model is validated and can be

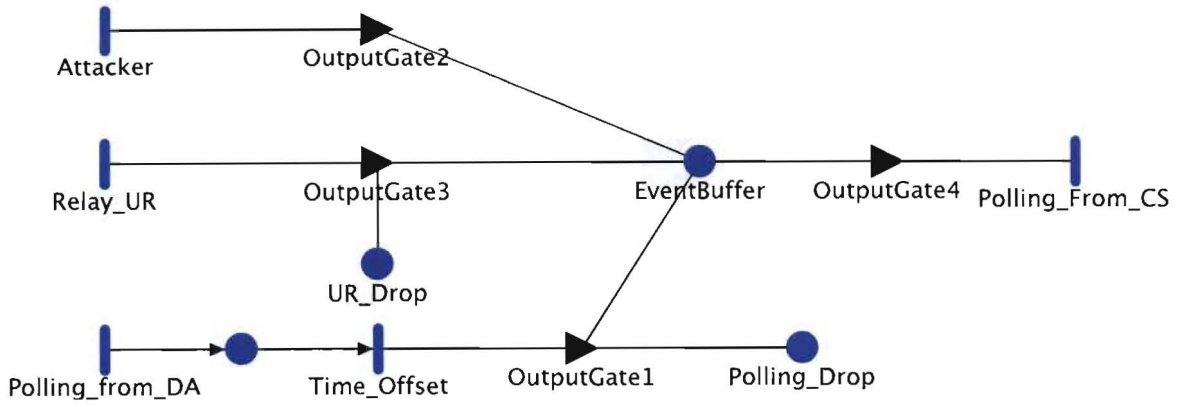

Figure 6. SAN Model of a DNP3-controlled Data Aggregator's Event Buffer in Möbius

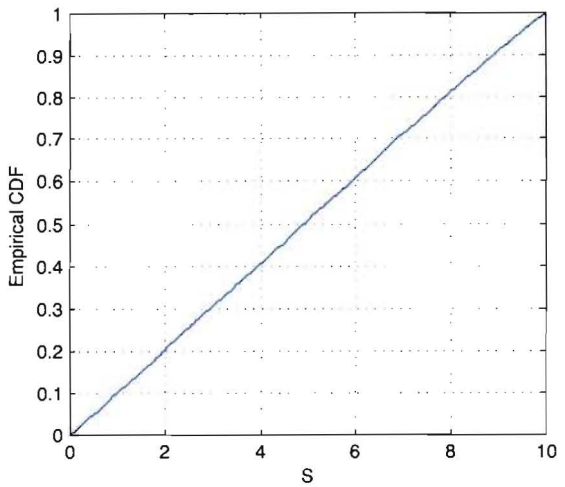

Figure 7. CDF of S: Time Difference Between Control Station's Poll And Data Aggregator's Poll

used for quantifying how the adversary's sending rate blocks legitimate traffic on the test data aggregator; furthermore, the simulation model can provide an accurate and flexible environment for exploring the model's parameter space for investigating event buffer flooding.

Table I
RELATIVE ERROR OF THE ESTIMATED FRACTION OF DROPPED (A) UNSOLICITED RESPONSE EVENTS AND (B) POLLING EVENTS FROM THE NORMAL RELAY

$\hat{y}$	y	Relative Error of Drop Fraction			
		UR Events		Polling Events	
		mean	std	mean	std
Analytical	Real	0.0245	0.0252	0.0535	0.0998
Simulation	Real	0.0206	0.0221	0.0494	0.0754
Analytical	Simulation	0.0056	0.0081	0.0105	0.0133

We observed that the test data aggregator simply sends everything inside the buffer in response to a control station's poll. If the number of events in the buffer is large, they will be fragmented into multiple DNP3 data packets that are resembled at the destination. Therefore, the real testbed has the constraint that $b = m$ and the corresponding DTMC model has only 1 state. However, it is recommended that in 2nd-level DNP3 slave, such as data aggregator in this case, the maximum number of items returned per poll be configurable in order to avoid overwhelming the network link [9]. Since the feature has been supported in many commercial data aggregators as well as by the Triangle Microworks' DNP3 testhareness [13], it is necessary to evaluate whether the analytical model correctly captures the attacker's effect on the data aggregator when $b > m$. The simulation model is used as a baseline to validate the analytical model. Let $m = 30$ and $b = 50$, now the DTMC model has 21 states. While keeping the rest parameters with the same values, we ran the same set of experiments on both the analytical model and the simulation model, and plot the unsolicited response events and polling events drop fractions in Figure 9(a) and 9(b) respectively. The drop fractions derived from the Möbius model are again the average of 10 independent runs with little variance. The relative error of the unsolicited response event drop fraction has mean of 0.0080 with standard deviation 0.0080, and the relative error of the polling event drop fraction has mean of 0.0066 with standard deviation of 0.0050. The extremely small relative error indicates that the DTMC model can efficiently compute the drop fraction of legitimate traffic as accurate as the simulation model.

D. Model Analysis

We then explore the impact on the drop fraction of key model parameters λ, w, S and m. The idea is to vary only one selected parameter for every set of experiments, and again measure the relationship between the attack sending rate and the fraction of dropped events. The baseline pa-

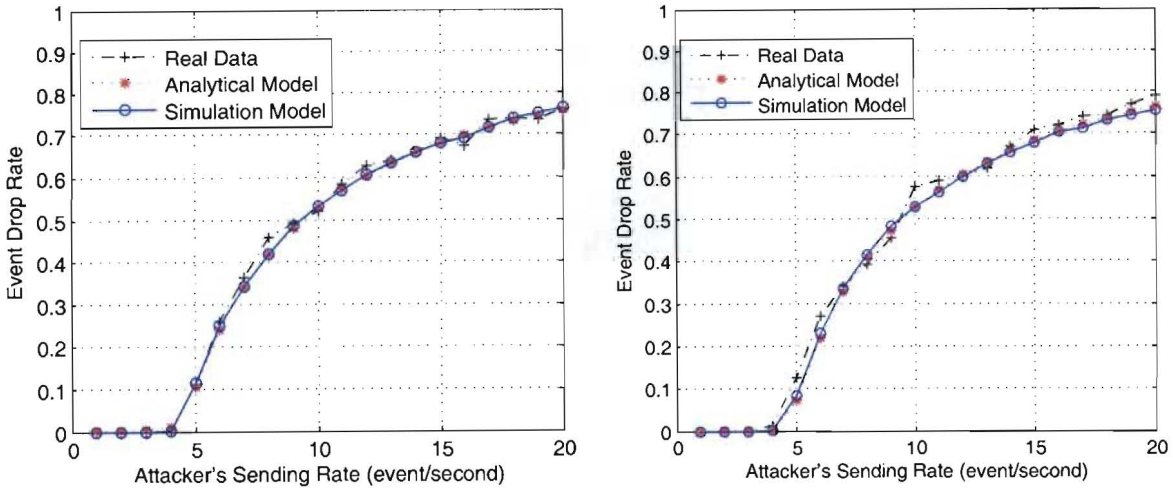

Figure 8. Estimated Fraction of Dropped (a) Unsolicited Response Events and (b) Polling Events from Normal Relay, Experimental Results from Real Testbed, Analytical Model and Simulation Model

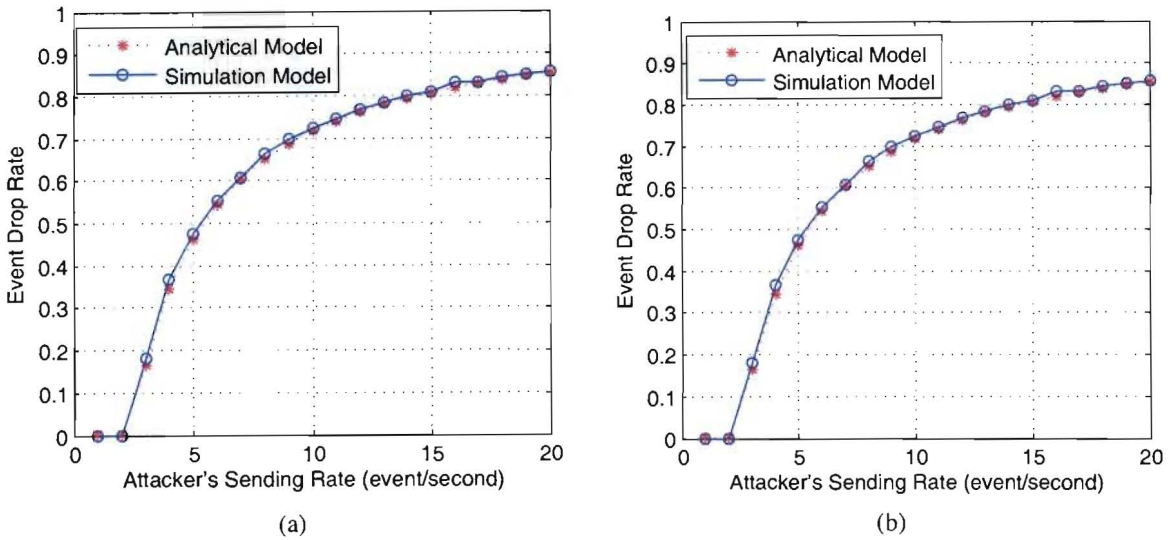

Figure 9. Estimated Fraction of Dropped (a) Unsolicited Response Events and (b) Polling Events from Normal Relay, with b = 50, m = 30

rameters are chosen as follows: $b = 50, m = 30, \delta = 10, \lambda = 0.3, w = 3, S$ is uniformly distributed between 0 and 10. Figure 10 displays the plots of drop fractions versus attacking rate for every selected parameter.

λ is the mean arrival rate of unsolicited response events from normal relay. Figure 10 (a1) and (a2) shows that all the lines with different λ values tend to converge as the attacker sending rate increases. Once the attacker sending rate is greater than 10 event per second, which is easy to achieve, λ has small impact on the both types of dropped events.

w is the number of events collected from the normal relay in response to a data aggregator's poll. Similar to the impact of λ, the lines tend to converge as attacker rate increases and thus w also has small impact on both types of event drop fractions, especially on the unsolicited response events.

S is the time offset between neighboring control station's poll and data aggregator's poll. The variation we noted earlier was taken over successive experiments. Under the assumption that both the control station polling is constant and that the data aggregator's polling is constant, in any given experiment S will be constant. We vary it here to see what impact a given constant S may have. It has little impact on the unsolicited response events. Within a polling interval, the number of attacking events is much more than the number of the normal relay's polling events, therefore

when the polling events arrive has minimum impact on the drop fraction of the unsolicited response events from the normal relay. However, the value of S greatly affects the fraction of polling events that are dropped. If the polling events arrive right after the previous control station's poll, there is always space in the buffer to hold them. On the other hand, if the polling events arrive just before the next control station's poll, the buffer has almost surely been filled up by the attacking events.

S varies in real because of the uncontrollable variance in the clocks that DNP3 masters use for issuing periodic polling requests. One enhancement could be developing rules on the data aggregator to generate polling requests to all the connected relays right after a control stations's poll (use multicast if supported), the polling events from normal relay can possibly enter the data aggregator's buffer before the attacking events overflow the buffer and minimize the fraction of dropped packets.

m is the maximum number of events transmitted to control station in response to a control station poll. Larger m essentially means larger service rate, and results in more available buffer space at the beginning of each time slot. Therefore, the fraction of dropped events of both types Frare reduced as shown in Figure 10 (d1) and (d2). However, increasing m is generally not a good solution, because the control station actually wastes even more resources including processing power and communication bandwidth to serve the attacking events. As a result, the adversary's impact effectively propagates to the communication between the control station and the data aggregator.

VII. Countermeasures

The key reason that event buffer flooding works is that buffer space is shared among sources, and use of the buffer follows a first-come-first-serve rule. The fraction of service that a data flow receives is always proportional to its input rate with FCFS policy when the buffer is congested. Therefore a high load flow like those of the attacker relay's unsolicited response events, can occupy most of the bandwidth, and influence the low load flows, such as the unsolicited response events and polling events from the normal relay. Another class of scheduling policies is designed with the goal of providing fair queueing [14], such as round robin (RR), weighted round robin (WRR) [15], weighted fair queueing [16] and virtual clock [17]. Applied in this context, the fair queueing scheduling policies aim to ensure that every input flow has reserved buffer space, and the additional buffer space will be equally distributed among flows that need more. Therefore, a reasonable defense against event buffer flooding is to allocate space in a shared event buffer according to a fair queueing policy. Round robin based scheduling could be a good choice due to the low time complexity O(1) and the low implementation cost [18].

As specified in the DNP3 protocol standard, every DNP3 slave's application response header contains a two-octet internal indications (IIN) field [9]. The bits in these two octets indicate certain states and error conditions within the slave. The third bit of the second octet indicates that an event buffer overflow condition exists in the DNP3 slave and at least one unconfirmed event was lost because the event buffers did not have enough room to store the information. The overflow condition continues to hold until the slave has available event buffer. It provides a means for the DNP3 master to detect whenever a buffer overflow occurs, however, the action recommended by the DNP3 user group, and in fact many vendors implemented in their products, is to issue an integrity poll in order to reestablish the current state of all data in the slave device [19]. However, the action is not sufficient to protect the device from event buffer flooding discussed in this paper. The integrity poll is passively issued upon receiving a response from DNP3 slave, and therefore it can only delay the time that next buffer overflow occurs. In addition, an integrity poll simply asks for all the static data rather than changed events, therefore generating many integrity polls could potentially overwhelm the network link between data aggregator and control station, and as a result, unintentionally wasting bandwidth and processing resources. One improvement could be applying rule-based policies to limit or filter the attacking traffic. For example, if relay A causes three successive sets of the event buffer overflow indication bit, the data aggregator will filter any data traffic whose DNP3 source address is of relay A. The rule will continue to take effect if the upcoming traffic from relay A exceeds a configured threshold. In addition, if the data aggregator's scheduling algorithm involves computation of weight, such as weighted round robin and weighted fair queueing, we could associate the event buffer overflow indication with an extremely small weight, and therefore minimizes amount of the attacking traffic entering the event buffer.

Lack of authentication in the DNP3 protocol enables adversaries to spoof normal relays. Researchers are actively working on various forms of crypto-based solutions to establish strong authentication in the SCADA environment, such as studying the practicality of various forms of key management [20], examining the practicality of using puzzle-based identification techniques to prevent DOS attack in a large scale network [21], or evaluating enhanced DNP3 protocols like DNP3 Secure Authentication [5] or DNPSec [6].

VIII. Related Work

DNP3 was designed without concern for security because SCADA networks were physically isolated with other networks at that time. However, with the growing of smart grid technologies, dependences of critical infrastructures on interconnected physical and cyber-based control systems grow, and so do vulnerabilities. The buffer flooding attack

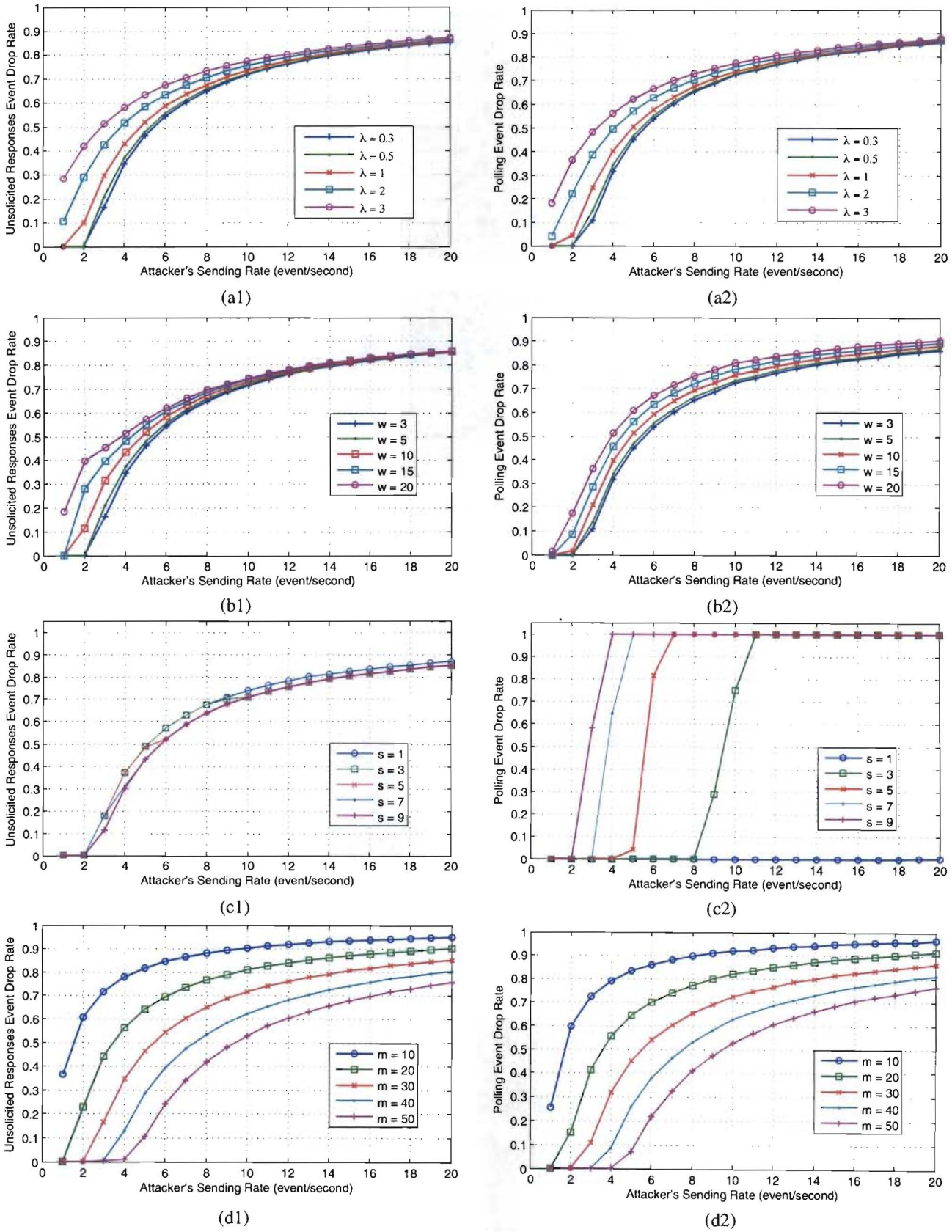

Figure 10. Model Analysis: Fraction of Dropped Unsolicited Response/Polling Events vs Attacking Sending Rate with varying (a) λ (b) w (c) S (d) m

discussed in this work targets data aggregators, and results in the loss of awareness in the control center. Detailed attacks against DNP3 specifications across all three layers were also proposed and classified into 28 generic attacks and 91 specific instances [7]. The impact of those attacks could result in loss of confidentiality, loss of awareness and even loss of control. A survey of SCADA-related attacks was conducted in [22], covering techniques of attack trees, fault trees, and risk analysis specific to critical infrastructures. The buffer flooding attack overwhelms the limited buffer resources in data aggregators, and thus it belongs to the class of DoS attacks. DoS attack and defense mechanisms in the Internet have been studied and classified in [23]. The real-time constraints and limited resources of the SCADA network makes the defense of such DoS attack even hard. Much research has also been done on realistic cyber attack vectors [24][25][26] and security gaps [27][28][29] specific to SCADA networks.

Investigation of attack vectors and security gaps will result in remediation techniques that can provide protection. Research has been done on countermeasures specific to DNP3 attacks, including data set security [30], SCADA-specific intrusion detection/prevention systems with sophisticated DNP3 rules [31] [32], and encapsulating DNP3 in another secure protocol such as SSL/TLS or IPSec [33]. Design guidances for authentication protocols based on extensive studies of the DNP3 Secure Authentication was proposed in [34]. Clearly, the smart grid technologies will bring much more attacks to the existing and new SCADA networks, therefore both independent researchers and government officials have formed workgroups [35] to investigate and offer their advice [36].

IX. Conclusion

This paper investigates how DNP3-controlled data aggregators respond to event buffer flooding. The adversary spoofs or captures a normal relay, and floods the connected data aggregator with unsolicited response events as if they are coming from the victim relay. The goal is to overload the shared event buffer in the data aggregator so that events from other normal relays will be dropped upon arriving to a full buffer. We study event buffer flooding on a real data aggregator. Also a DTMC model and a Möbius simulation model have been developed for analyzing its behavior.

Acknowledgment

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

References

[1] G. Andersson, P. Donalek, R. Farmer, N. Hatziargyriou, I. Kamwa, P. Kundur, N. Martins, J. Paserba, P. Pourbeik, J. Sanchez-Gasca *et al.*, "Causes of the 2003 major grid blackouts in North America and Europe, and recommended means to improve system dynamic performance," *Power Systems, IEEE Transactions on*, vol. 20, no. 4, pp. 1922–1928, 2005.

[2] (2010) Pacific Northwest National Laboratory (PNNL) Looking back at the august 2003 blackout. [Online]. Available: http://eioc.pnl.gov/research/2003blackout.stm

[3] S. Corsi and C. Sabelli, "General blackout in italy sunday september 28, 2003, h. 03: 28: 00," in *Power Engineering Society General Meeting, 2004. IEEE.* IEEE, 2005, pp. 1691–1702.

[4] (2008) Electric Power Research Insititute, "DNP security development, evaluation and testing project opportunity". [Online]. Available: http://mydocs.epri.com/docs/public/000000000001016988.pdf

[5] (2010, March) DNP Users Group, "dnp3 specification, secure authentication, supplement to volume 2". [Online]. Available: http://www.dnp.org/Modules/Library/Document.aspx

[6] M. Majdalawieh, F. Parisi-Presicce, and D. Wijesekera, "DNPSec: Distributed Network Protocol Version 3 (DNP3) Security Framework," *Advances in Computer, Information, and Systems Sciences, and Engineering*, pp. 227–234, 2006.

[7] S. East, J. Butts, M. Papa, and S. Shenoi, "A Taxonomy of Attacks on the DNP3 Protocol," *Critical Infrastructure Protection III*, pp. 67–81, 2009.

[8] Möbius Team, *The Möbius Manual www.mobius.illinois.edu*, University of Illinois, Urbana Champaign, Urbana, IL, 2010.

[9] *DNP3 Specification Application Layer Volume 2 Part 1*, 2007.

[10] J. Meyer, A. Movaghar, and W. Sanders, "Stochastic activity networks: Structure, behavior, and application," in *International Workshop on Timed Petri Nets.* IEEE Computer Society, 1985, pp. 106–115.

[11] W. Sanders, "Integrated frameworks for multi-level and multi-formalism modeling," in *Petri Nets and Performance Models, 1999. Proceedings. The 8th International Workshop on.* IEEE, 2002, pp. 2–9.

[12] F. Bause and P. Kritzinger, *Stochastic Petri Nets*. Vieweg, 2002.

[13] Triangle Microworks. (2009) Dnp3 communication protocol test harness. [Online]. Available: http://www.trianglemicroworks.com/

[14] D. Stiliadis and A. Varma, "Design and analysis of frame-based fair queueing: A new traffic scheduling algorithm for packet-switched networks," *SIGMETRICS Performance Evaluation Review*, vol. 24, no. 1, pp. 104–115, 1996.

[15] M. Katevenis, S. Sidiropoulos, and C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip," *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 8, pp. 1265–1279, 1991.

[16] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Symposium proceedings on Communications architectures & protocols*. ACM, 1989, pp. 1–12.

[17] L. Zhang, "VirtualClock: A new traffic control algorithm for packet-switched networks," *ACM Transactions on Computer Systems (TOCS)*, vol. 9, no. 2, p. 124, 1991.

[18] C. Guo, "SRR: An O(1) time complexity packet scheduler for flows in multi-service packet networks," in *SIGCOMM '01: Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 2001, pp. 211–222.

[19] *DNP3 Specification Application Layer Volume 2 Part 2*, 2007.

[20] L. Pièrre-Cambacédès and P. Sitbon, "Cryptographic key management for SCADA systems-issues and perspectives," in *Information Security and Assurance, 2008. ISA 2008. International Conference on*. IEEE, 2008, pp. 156–161.

[21] C. Bowen III, T. Buennemeyer, and R. Thomas, "A Plan for SCADA Security Employing Best Practices and Client Puzzles to Deter DoS Attacks," *Working Together: R&D Partnerships in Homeland Security*, 2005.

[22] P. Ralston, J. Graham, and J. Hieb, "Cyber security risk assessment for SCADA and DCS networks," *ISA transactions*, vol. 46, no. 4, pp. 583–594, 2007.

[23] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[24] E. Bompard, C. Gao, R. Napoli, A. Russo, M. Masera, and A. Stefanini, "Risk assessment of malicious attacks against power systems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 5, pp. 1074–1085, 2009.

[25] J. Fernandez and A. Fernandez, "SCADA systems: vulnerabilities and remediation," *Journal of Computing Sciences in Colleges*, vol. 20, no. 4, pp. 160–168, 2005.

[26] V. Igure, S. Laughter, and R. Williams, "Security issues in SCADA networks," *Computers & Security*, vol. 25, no. 7, pp. 498–506, 2006.

[27] S. Patel and Y. Yu, "Analysis of SCADA Security models," *International Management Review*, vol. 3, no. 2, 2007.

[28] A. Faruk, "Testing & Exploring Vulnerabilities of the Applications Implementing DNP3 Protocol," 2008.

[29] S. Hong and S. Lee, "Challenges and pespectives in security measures for the SCADA system," in *Proc. 5th Myongji-Tsinghua University Joini Seminar on Prototection & Automation*, 2008.

[30] T. Mander, R. Cheung, and F. Nabhani, "Power system DNP3 data object security using data sets," *Computers & Security*, vol. 29, no. 4, pp. 487–500, 2010.

[31] D. Rrushi and U. di Milano, "SCADA Intrusion Prevention System," in *Proceedings of 1st CI2RCO Critical Information Infrastructure Protection Conference*, 2006.

[32] (2010) Digital Bond, DNP3 IDS Signatures. [Online]. Available: http://www.digitalbond.com/index.php/research/scada-idsips/ids-signatures/dnp3-ids-signatures-2/

[33] J. Graham and S. Patel, "Security considerations in SCADA communication protocols," Intelligent Systems Research Laboratory, Tech. Rep. TR-ISRL-04-01, 2004.

[34] H. Khurana, R. Bobba, T. Yardley, P. Agarwal, and E. Heine, "Design Principles for Power Grid Cyber-Infrastructure Authentication Protocols," in *43th Annual Hawaii International Conference on System Sciences*, 2010.

[35] (2010) US National Institute of Standards and Technology (NIST), smart grid interoperability standards project. [Online]. Available: http://www.nist.gov/smartgrid/

[36] (2010, August) US National Institute of Standards and Technology (NIST), guidelines for smart grid cyber security. [Online]. Available: http://csrc.nist.gov/publications/PubsNISTIRs.html#NIST-IR-7628