Proceedings of the 2011 Winter Simulation Conference
S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, eds.

AN EVENT BUFFER FLOODING ATTACK IN DNP3 CONTROLLED SCADA SYSTEMS

Dong Jin
David M. Nicol

University of Illinois at Urbana-Champaign
Urbana, IL, 61801, USA

Guanhua Yan

Los Alamos National Laboratory
Los Alamos, NM 87545, USA

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 receives observed state from devices within a local region, and the control center collects the aggregated state from the data aggregator. The DNP3 communications are asynchronous across the two levels; this leads to the possibility of completely filling a data aggregator's buffer of pending events, when a compromised relay sends overly many (false) events to the data aggregator. This paper investigates the attack by implementing the attack using real SCADA system hardware and software. A Discrete-Time Markov Chain (DTMC) model is developed for understanding conditions under which the attack is successful and effective. The model is validated by a Möbius simulation model and data collected on a real SCADA testbed.

1 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 (PNNL 2010); the obvious threat is that attackers harm the grid infrastructure through largely electronic means.

We are interested in a vulnerability that arises within the communication infrastructure of the grid. The Distributed Network Protocol v3.0 (DNP3) is the most widely used SCADA network communication protocol in North America (approximately 75%) (EPRI 2008). Designed to provide interoperability and as an open standard to device manufacturers, 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 (DNP Users Group 2010) or DNPSec (Majdalawieh, Parisi-Presicce, and Wijesekera 2006), 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 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 (East, Butts, Papa, and Shenoi 2009). The attack we identified in this paper is against the vendor implementation as well as the underlying communication structure. An attacker on the network can simply send many data events to a device that temporarily buffers SCADA data before they are retrieved by a control station. The attack fills an event buffer so as to prohibit the buffering of critical alerts from legitimate devices, negatively impacting the control station's *situational awareness*. The simple attack works effectively because (1) many commercial DNP3 data aggregators implement shared event buffer and (2) the communication between a control center and a data aggregator is asynchronous with the communication between a data aggregator and relays. In addition, many proof-based Denial of

Service (DoS) defense techniques, such as client-puzzle and public/private key, may not work appropriately because SCADA networks are generally resource-limited and have strong real-time requirements.

In a nutshell, the main contributions of this paper are: (1) we identify a simple but very effective flooding attack in DNP3-controlled SCADA networks. We prove the existence and effectiveness of the attack using commercial power grid equipment in our lab; (2) we develop a DTMC model for analyzing the effectiveness of the attack as a function of various behavioral parameters. The analytical model has been validated by the data from the real testbed and a simulation model created in Möbius (Möbius 2010); (3) we suggest some countermeasures against this type of attacks.

The remainder of the paper is organized as follows: Section 2 gives an overview of DNP3-controlled SCADA networks. Section 3 describes the threat model. Section 4 introduces the vulnerabilities in DNP3 slave devices used by this attack. Section 5 explores the existence of the buffer flooding attack on a real data aggregator. Section 6 presents a DTMC model and a simulation model of the attack, and compares the two models with results from the real data aggregator. Section 7 discusses countermeasures against this type of attack and Section 8 describes the related work. Finally, we draw concluding remarks in Section 9.

2 DNP3 OVERVIEW

The DNP3 protocol carries on control and data communication among SCADA system components. It is a master-slave based protocol. 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 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. An 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. 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* can spontaneously send events to its master. This is useful for reporting state changes where a reaction is time-critical. The attack we have identified exploits the unsolicited response mode.

3 THREAT MODEL

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 are the data aggregators within 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 attackers must establish a connection with the data

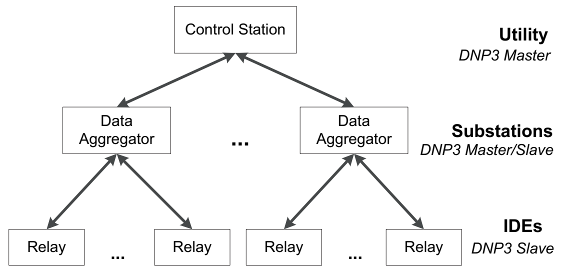

Figure 1: Two-level architecture of a DNP3-controlled SCADA network.

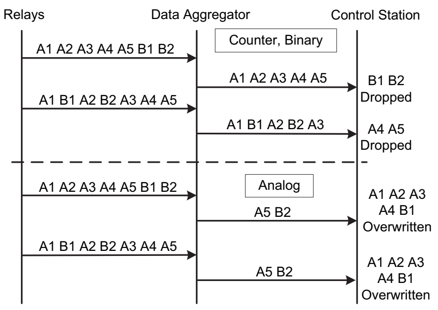

Figure 2: Time Sequence Diagram for Revealing Data Aggregator's Buffering Mechanism with Buffer Size = 5.

aggregator as a legitimate 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 attackers from spoofing the relays. The attackers 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 establish a new connection with the data aggregator. The attackers 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. 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 listed at the top of an attacker'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 (Shaw 2006). Furthermore, most relays do not have a limit on the number of login attempts, which could easily make a typical automated password cracker software effective.

4 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 fed by all slaves from which the data aggregator acquires data. The attack has a compromised or spoofed DNP3 slave send so many unsolicited events that the buffer is filled, and events from legitimate slaves are lost until the buffer is emptied by a query from the control station.

5 EXPERIMENTS ON DATA AGGREGATOR

5.1 Buffering Mechanism Experiments

The DNP3 specification describes the general guidelines on event buffer semantics and leaves the implementation to vendors (DNP Users Group 2007). The vendor's implementation is generally not publicly available. Therefore, in order to verify the existence of this buffer flooding attack, 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 mentioned in Section 2 (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 the buffer flooding attack. 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 attacker's flooding affects only the buffer space allocated for the attacker's device.

5.2 Buffer Flooding Experiments

The next experiment launches buffer flooding attacks. The data aggregator serves as a 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 attacker and it can generate many unsolicited response events and stop responding to polling requests. The unsolicited response event traffic from the attacker 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) for 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 attacker 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 unsolicited 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.

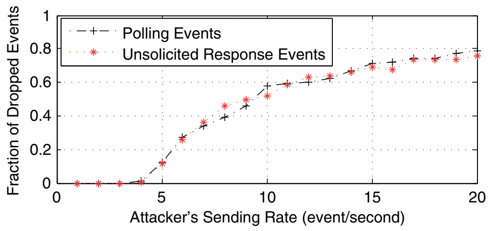

Figure 3: Fraction of Dropped Events from Normal Relay on Real Testbed.

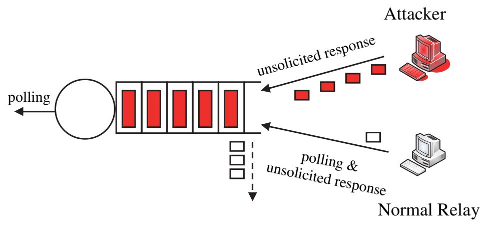

Figure 4: Queueing Diagram of the Data Aggregator's Event Buffer.

6 MODELING AND ANALYSIS

We develop a DTMC analytical model and a Möbius simulation model for investigating this buffer flooding attack. Both models grant us the flexibility and controllability to explore the influence of various behavioral parameters, some of which are hard to configure in real testbed. Once we well understand the small-scale model (Figure 4) in this paper, we plan to conduct experiments on large-scale models of real utility SCADA systems as the next step.

6.1 Analytical Model

The DTMC state is the buffer size at the instant a control station poll request arrives. The time-step is the control station polling interval length. Figure 4 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 5 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	attacker's unsolicited response event sending rate, events arrive in constant bit rate
λ	mean arrival rate of unsolicited response events from normal relay (poisson arrival)
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 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 $Q(k+1) = [min(Q(k)+A(k),b)-D(k)]^+$. 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

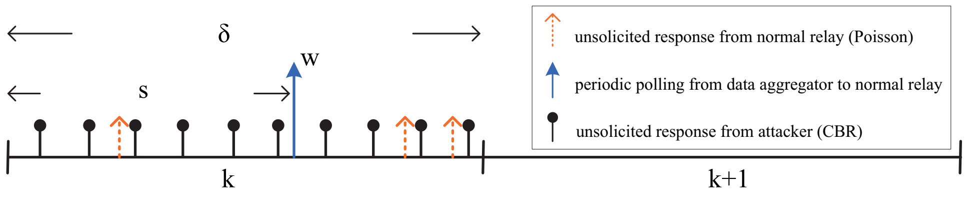
δ
w
s
k
k+1
unsolicited response from normal relay (Poisson)
periodic polling from data aggregator to normal relay
unsolicited response from attacker (CBR)

Figure 5: Timing Diagram of Event Arrivals.

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

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

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. The dependence on k is removed from the notation of the distribution of A as we are interested in the asymptotic behavior. 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, and $L_i=((A-(b-i))^+$. The average number of dropped events per time slot is computed as $E(L)=\sum_{i=0}^{b}\pi_i E[(A-(b-i))^+]$. The ratio of expected dropped events of all types to expected events in a time slot is $\rho=\frac{E(L)}{E(A)}=\frac{E(L)}{r\delta+\lambda\delta+w}$, a value which by Jensen's Inequality (Ross 1996) 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.

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

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

The average number of dropped unsolicited response events and polling events from normal relay given T_f can be computed respectively as $E(L_i^{ur}|T_f=t,S=s)=E(L_i^{ur}|T_f=t)=(\delta-t)\lambda$, and

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

The average number of dropped unsolicited response events and polling events from normal relay within a time slot can be derived respectively as $E(L^{ur})=\sum_{i=0}^{b-m}\pi_i\int_{s=0}^{\delta}f(s)\sum_t\bar{P}_i(T_f=t|S=s)E(L_i^{ur}|T_f=t,S=s)ds$ and $E(L^{poll})=\sum_{i=0}^{b-m}\pi_i\int_{s=0}^{\delta}f(s)\sum_t\bar{P}_i(T_f=t|S=s)E(L_i^{poll}|T_f=t,S=s)ds$, where $P_i(T_f=t|S=s)$ is normalized by $\bar{P}_i(T_f=t|S=s)=\frac{P_i(T_f=t|S=s)}{\sum_t P_i(T_f=t|S=s)}$. Thus, a lower bound on the expected fraction of lost normal unsolicited response events is $\rho^{ur}=\frac{E(L^{ur})}{\lambda\delta}$, while the exact expected fraction of lost normal polling events is $\rho^{poll}=\frac{E(L^{poll})}{w}$. ρ^{poll} is exact because w is constant in this model.

6.2 Simulation Model

We also built a stochastic activity network (SAN) (Meyer, Movaghar, and Sanders 1985) simulation model with respect to the real testbed setup in Möbius v2.3.1, which provides a flexible, extensible, and efficient framework for implementing algorithms to model and solve discrete-event systems. SANs consist of four primitive objects: places, activities, input gates, and output gates (Möbius 2010). Figure 6 shows the core design of the event buffer attack 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.

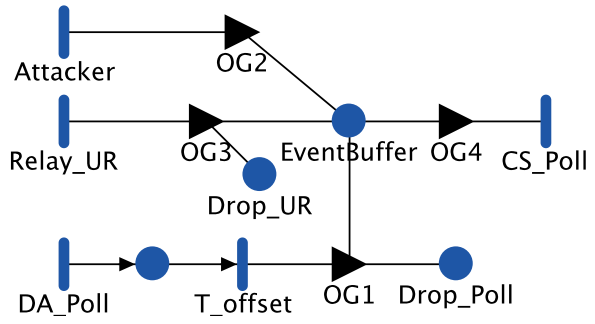

Figure 6: SAN Model of a DNP3-controlled Data Aggregator's Event Buffer in Möbius.

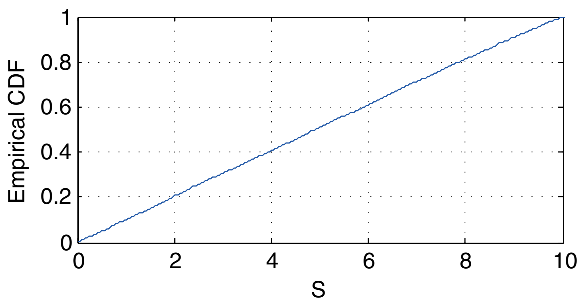

Figure 7: CDF of S: Time Difference Between Control Station's Poll And Data Aggregator's Poll.

6.3 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 1. 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 used for quantifying how the attacker'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 the buffer flooding attack.

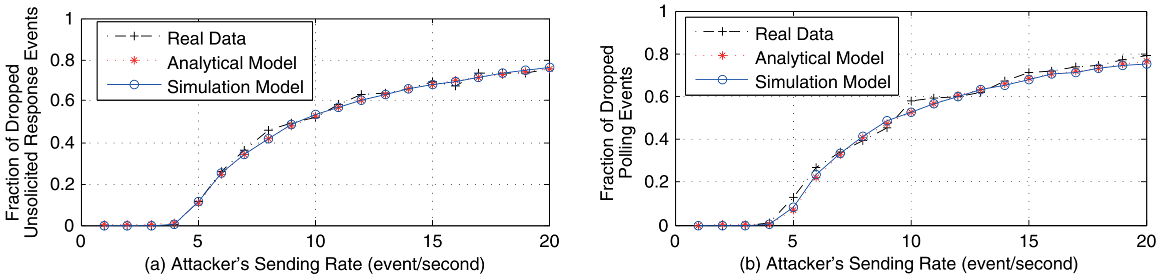

Figure 8: Estimated Fraction of Dropped (a) Unsolicited Response Events and (b) Polling Events from Normal Relay, Experimental Results from Real Testbed, Analytical Model and Simulation Model.

Table 1: Relative Error of the Estimated Fraction of Dropped 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 (DNP Users Group 2007). Since the feature has been supported in many commercial data aggregators, 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 small relative error indicates that the DTMC model can efficiently compute the drop fraction of legitimate traffic as accurate as the simulation model.

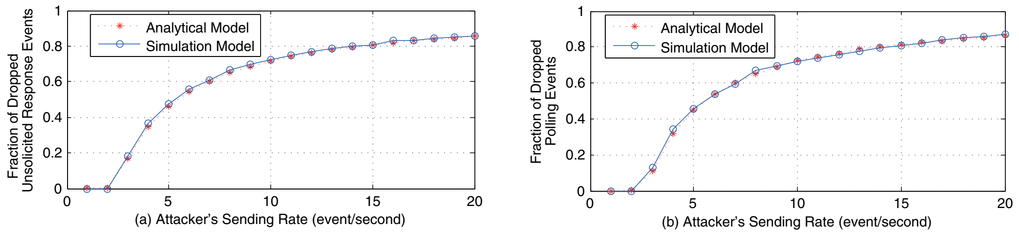

Figure 9: Estimated Fraction of Dropped (a) Unsolicited Response Events and (b) Polling Events from Normal Relay, with b = 50, m = 30.

6.4 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 parameters 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 and 11 displays the plots of drop fractions versus attacking rate for every selected parameter.

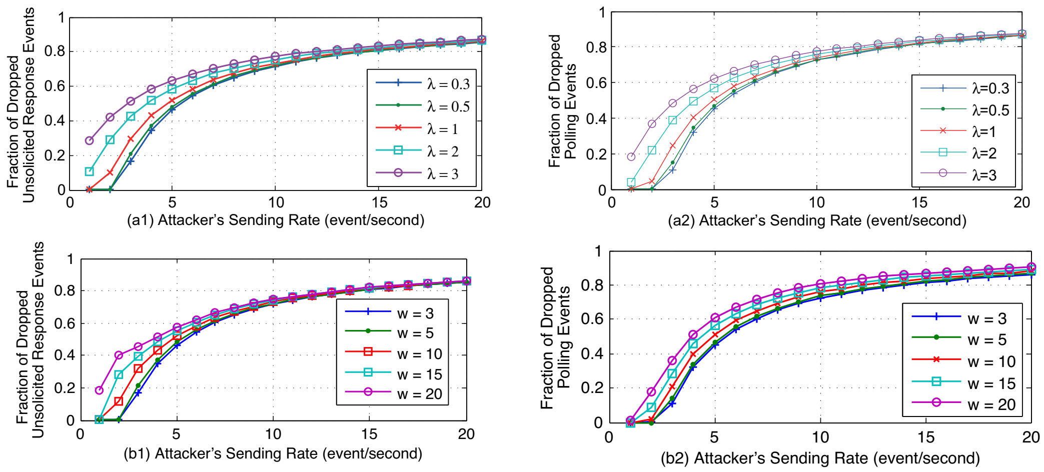

Figure 10: Fraction of Dropped Events vs Attacking Sending Rate with varying (a) λ (b) w.

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 reality 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 station'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 fractions of dropped events of both types are reduced as shown in Figure 11 (b1) and (b2). 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 attacker's impact effectively propagates to the communication between the control station and the data aggregator.

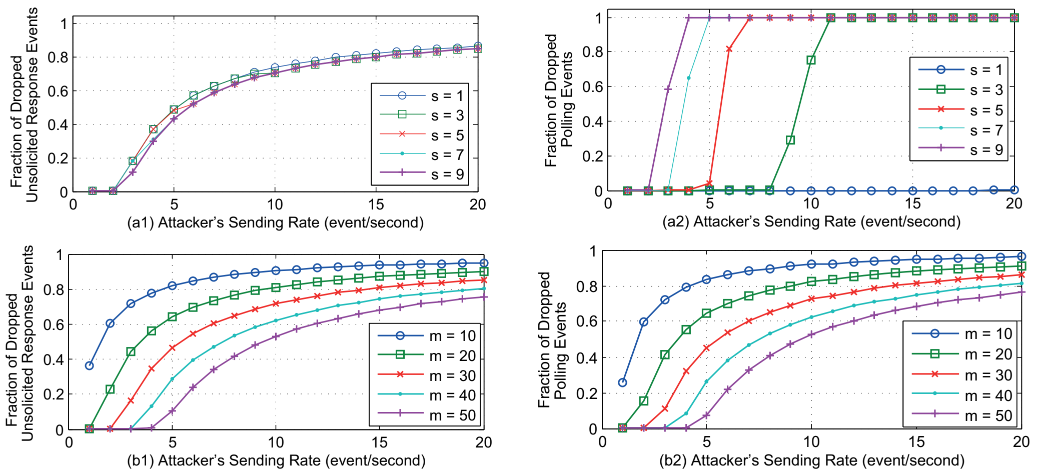

Figure 11: Fraction of Dropped Events vs Attacking Sending Rate with varying (a) *S* (b) *m*.

7 COUNTERMEASURES

The key reason that the buffer flooding attack works is that the buffer space is shared among multiple 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 (Stiliadis and Varma 1996), such as round robin (RR), weighted round robin (WRR) and weighted fair queueing. 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 the buffer flooding attack 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 (Guo 2001).

As specified in the DNP3 protocol standard, every DNP3 slave's application response header contains a two-octet internal indications (IIN) field (DNP Users Group 2007). 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. 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 (DNP Users Group 2007). However, the action is not sufficient to protect the device from the flooding attack 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 the amount of attacking traffic entering the event buffer.

Lack of authentication in the DNP3 protocol enables attackers 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 (Piètre-Cambacédès and Sitbon 2008), examining the practicality of using puzzle-based identification techniques to prevent DOS attack in a large scale network (Bowen III, Buennemeyer, and Thomas 2005), or evaluating enhanced DNP3 protocols like DNP3 Secure Authentication (DNP Users Group 2010) or DNPSec (Majdalawieh, Parisi-Presicce, and Wijesekera 2006).

8 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 attack discussed in this work targets data aggregators, and results in the loss of situational 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 (East, Butts, Papa, and Shenoi 2009). 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 (Ralston, Graham, and Hieb 2007), 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 (Mirkovic and Reiher 2004). The real-time constraints and limited resources of the SCADA network makes the defense of such DoS attack even harder. Much research has also been done on realistic cyber attack vectors and security gaps specific to SCADA networks (Fernandez and Fernandez 2005, Faruk 2008).

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 (Mander, Cheung, and Nabhani 2010), SCADA-specific intrusion detection/prevention systems with sophisticated DNP3 rules (Bond 2010), and encapsulating DNP3 in another secure protocol such as SSL/TLS or IPSec (Graham and Patel 2004). Design guidances for authentication protocols based on extensive studies of the DNP3 Secure Authentication was proposed in Khurana, Bobba, Yardley, Agarwal, and Heine (2010).

9 CONCLUSION

This paper investigates a buffer flooding attack on DNP3-controlled data aggregators. The attacker 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. The attack has been implemented on a real data aggregator. Also a DTMC model and a Möbius simulation model have been developed for analyzing the behavior of such attacks. Results have shown the simple flooding attack can be very effective, and strong authentication is definitely required towards securing the DNP3-controlled SCADA networks.

ACKNOWLEDGMENTS

This material is based upon work supported by the Department of Energy under Award Number DE-OE0000097. The views expressed are those of the authors only.

REFERENCES

Digital Bond 2010. “DNP3 IDS Signatures”. http://www.digitalbond.com.

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

DNP Users Group 2007. *DNP3 Specification Application Layer Volume 2*. DNP Users Group.

DNP Users Group 2010, March. “DNP3 Specification, Secure Authentication, Supplement to Volume 2”. http://www.dnp.org/Modules/Library/Document.aspx.

East, S., J. Butts, M. Papa, and S. Shenoi. 2009. “A Taxonomy of Attacks on the DNP3 Protocol”. *Critical Infrastructure Protection III* 311:67–81.

EPRI 2008. “DNP Security Development, Evaluation and Testing Project Opportunity”. http://mydocs.epri.com/docs/public/000000000001016988.pdf.

Faruk, O. 2008. “Testing & Exploring Vulnerabilities of the Applications Implementing DNP3 Protocol”.

Fernandez, J., and A. Fernandez. 2005. “SCADA systems: vulnerabilities and remediation”. *Journal of Computing Sciences in Colleges* 20.

Graham, J., and S. Patel. 2004. “Security considerations in SCADA communication protocols”. Technical Report TR-ISRL-04-01, Intelligent Systems Research Laboratory.

Guo, C. 2001. “SRR: An O(1) time complexity packet scheduler for flows in multi-service packet networks”. In *ACM SIGCOMM*.

Khurana, H., R. Bobba, T. Yardley, P. Agarwal, and E. Heine. 2010. “Design Principles for Power Grid Cyber-Infrastructure Authentication Protocols”. In *HICSS*, 1–10.

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

Mander, T., R. Cheung, and F. Nabhani. 2010. “Power system DNP3 data object security using data sets”. *Computers & Security* 29 (4): 487–500.

Meyer, J., A. Movaghar, and W. Sanders. 1985. “Stochastic activity networks: Structure, behavior, and application”. In *International Workshop on Timed Petri Nets*.

Mirkovic, J., and P. Reiher. 2004. “A taxonomy of DDoS attack and DDoS defense mechanisms”. *ACM SIGCOMM Computer Communication Review* 34 (2): 39–53.

Möbius 2010. “The Möbius Manual”. www.mobius.illinois.edu.

Piètre-Cambacédès, L., and P. Sitbon. 2008. “Cryptographic key management for SCADA systems-issues and perspectives”. In *Information Security and Assurance*.

PNNL 2010. “Looking Back at the August 2003 Blackout”. http://eioc.pnl.gov/research/2003blackout.stm.

Ralston, P., J. Graham, and J. Hieb. 2007. “Cyber security risk assessment for SCADA and DCS networks”. *ISA transactions* 46 (4): 583–594.

Ross, S. 1996. *Stochastic processes*. Wiley New York.

Shaw, W. 2006. *Cybersecurity for SCADA systems*. Pennwell Corp.

Stiliadis, D., and A. Varma. 1996. “Design and analysis of frame-based fair queueing: A new traffic scheduling algorithm for packet-switched networks”. *SIGMETRICS Performance Evaluation Review*.

AUTHOR BIOGRAPHIES

DONG JIN is a Ph.D. student of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He holds a B.Eng. with first class honors in computer engineering from Nanyang Technological University (2005), and a M.S. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (2010). His research interests lie in the areas of computer security, large-scale computer system modeling and simulation. His email address is dongjin2@illinois.edu.

DAVID M. NICOL is Professor of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. He holds a B.A. in mathematics from Carleton College (1979), and M.S. and Ph.D. degrees in computer science from the University of Virginia (1983,1985). Prior to joining UIUC, he taught at the College of William & Mary, and Dartmouth College. He has served in many roles in the simulation community (*e.g.*, Editor-in-Chief of ACM TOMACS, General Chair of the Winter Simulation Conference Executive Board of the WSC), was elected Fellow of the IEEE and Fellow of the ACM for his work in discrete-event simulation, and was the inaugural recipient of the ACM SIGSIM Distinguished Contributions award. His current research interests include application of simulation methodologies to the study of security in computer and communication systems. His email address is dmnicol@illinois.edu.

GUANHUA YAN is a Technical Staff Member in the Information Sciences Group (CCS-3) at Los Alamos National Laboratory. He holds a Ph.D. degree in Computer Science from Dartmouth College, USA, in 2005. From 2003 to 2005, he was a visiting graduate student at the Coordinated Science Laboratory in the University of Illinois at Urbana-Champaign. His current research interests are cyber modeling and simulation, anomaly detection, infrastructure protection, and data privacy. His email address is ghyan@lanl.gov.