Semantic Security Analysis of SCADA Networks to Detect Malicious Control Commands in Power Grids

Hui Lin[1], Adam Slagell[3], Zbigniew Kalbarczyk[1], Peter W. Sauer[2], Ravishankar K. Iyer[1]
[1]Coordinated Science Laboratory,
[2]Department of Electrical and Computer Engineering,
[3]National Center for Supercomputing Applications,
University of Illinois at Urbana-Champaign,
{hlin33, slagell, kalbarcz, psauer, rkiyer}@illinois.edu

ABSTRACT
In the current generation of SCADA (*S*upervisory *C*ontrol *A*nd *D*ata *A*cquisition) systems used in power grids, a sophisticated attacker can exploit system vulnerabilities and use a legitimate maliciously crafted command to cause a wide range of system changes that traditional contingency analysis does not consider and remedial action schemes cannot handle. To detect such malicious commands, we propose a semantic analysis framework based on a distributed network of intrusion detection systems (IDSes). The framework combines system knowledge of both cyber and physical infrastructure in power grid to help IDS to estimate execution consequences of control commands, thus to reveal attacker's malicious intentions. We evaluated the approach on the IEEE 30-bus system. Our experiments demonstrate that: (i) by opening 3 transmission lines, an attacker can avoid detection by the traditional contingency analysis and instantly put the tested 30-bus system into an insecure state and (ii) the semantic analysis provides reliable detection of malicious commands with a small amount of analysis time.

Categories and Subject Descriptors
K.6.5 [**Security and Protection**]

General Terms
Security

Keywords
SCADA, intrusion detection system, semantic analysis, contingency analysis

1. INTRODUCTION
Today's power grid relies on classical state estimation methods to obtain system states. To evaluate how a system state changes in case of an incident (e.g., a line outage), contingency analysis is usually performed. Because of time constraints, such evaluations are performed for low-order incidents, e.g., the "*N-1*" contingency evaluates incidents when a failure of one physical component is considered. In the current generation of SCADA (*S*upervisory *C*ontrol *A*nd *D*ata *A*cquisition) systems, a sophisticated attacker can exploit system vulnerabilities and use a single maliciously crafted control command to cause a wide range of system changes that traditional contingency analysis does not consider and remedial action schemes cannot handle [2].

It is challenging to detect such control-related attacks based solely on the state of physical components in the power system, mainly for two reasons. *First*, a power system usually periodically collects measurements and estimates its state accordingly, with an interval on the order of minutes. In this scenario, the system may detect the consequences of an attack based on the most current measurements, but this may be after the physical damage has occurred. *Second*, measurements can be compromised in the cyber environment in order to make the physical consequences of an attack invisible to control centers.

Detection of such attacks is also challenging if we solely rely on network intrusion detection systems (IDSes), even if IDSes are specifically designed for SCADA systems [3]. Maliciously crafted control commands are usually transmitted in a legitimate format and do not generate abnormal network traffic patterns. Furthermore, network semantics related to control commands cannot reveal attackers' malicious intentions.

In this paper, we propose a semantic analysis framework based on a distributed network of intrusion detection systems (IDSes) to detect malicious commands carried in legitimate formats. The proposed framework combines system knowledge of both cyber and physical infrastructure in power grid to estimate what the system state will be if a control command is allowed to execute, e.g., opening a circuit breaker in a substation. For each IDS instance, we have designed a dedicated network analyzer and integrated it with the Bro intrusion detection system [3]. The analyzer enables the IDS to focus on every network packet that carries critical control commands, e.g., commands that may operate physical devices. The network IDS triggers run-time power flow analysis based on the information extracted from the network packets, to estimate the physical consequences.

As most proprietary SCADA protocols still lack security properties to guarantee the integrity of network packets, distributed IDS instances are connected together to maintain a trusted cyber environment. In the proposed framework, each IDS instance is responsible for: (i) collecting trusted information locally, e.g., in the context of each substation and (ii) communicating with other IDS instances (e.g., an IDS instance at the control center) through secure channels to guarantee that the collected information is not tampered with. Deploying the trusted cross-IDS communication allows us to perform the semantic analysis at any physical location, and thus, it is possible to detect malicious control commands once they are issued.

Each IDS instance is based on Bro with integrated DNP3 analyzer (we recently developed [3]) and the framework is evaluated using the IEEE 30-bus test system. Our experiments demonstrate that: (i) by opening 3 transmission lines (possibly with a single

SEGS'13, November 08 2013, Berlin, Germany.

http://dx.doi.org/10.1145/2516930.2516947

command), an attacker can avoid detections by the traditional contingency analysis and instantly put the tested 30-bus system into an insecure state and (ii) the semantic analysis provides reliable monitoring and detection of malicious commands with a small amount of time.

The remainder of this paper is organized as follows. In Section 2, we introduce the threat model and an attack scenario. Section 3 presents the semantic analysis framework to detect malicious commands. Experimental evaluation of the proposed framework is described in Section 4. Section 5 discusses the related work and we conclude in Section 6.

2. ATTACK MODEL

In the traditional power grid, radio links or serial links were commonly used for communication within SCADA systems. However, in the current generation power systems, IP-based network infrastructure and intelligent devices are commonly being deployed to enable more accurate and efficient control at less cost. In this work, we consider the following threat model:

- ***We do not trust*** any "intelligent" component, including operation environment in *control centers*, automatic *field devices* in substations, and *control networks*. As most proprietary SCADA protocols lack security properties (e.g., authentication and integrity), we assume that an attacker can install a malicious software on the SCADA system to either directly observe real-time measurements arriving from substations [4], or access data historians. Based on the measurements, it is also possible for an attacker to estimate network topology and system states [5][6].
- ***We trust*** measurements of power usage, current, and voltage directly obtained from sensing devices (or sensors) in substations. As indicated in [7], performing false data injection attacks in power system requires: (i) manipulation of measurement data before they are used for state estimation or (ii) physical tampering with sensing devices. While we do not make any assumptions about the trustworthiness of data or devices upstream of sensors, we trust the information (voltage, current, etc.) at the sensors. Concurrent physical accesses to and tampering with a large number of distributed sensors (across multiple substations) is hard to achieve in practice. Also, as indicated in [8], it is sufficient to protect "a strategically selected set of sensor measurements" to detect false data injection attacks.
- ***We trust*** the functionality of IDSes. We consider a passive IDS (in the current implementation, IDS is merely responsible for detecting an intruder) and even the IDS compromise does not degrade the security of the power system.

2.1 Attack Scenario

We are concerned with attack scenarios in which an attacker: (i) can penetrate the intelligent component in the power system (e.g., see [1][4]) and (ii) can issue control commands to enforce malicious system changes. Specifically, we focus on malicious control commands that can put the power system into an insecure state instantly in a manner that cannot be handled via traditional contingency analysis and remedial action schemes [2].

Figure 1 shows an attack scenario (represented as a sequence of steps) to demonstrate a possible penetration procedure.

Entry Points. An attacker may penetrate a control center or field devices in substations as an insider or by remote accesses (e.g., by exploiting vulnerable software).

Attack Preparation Stage. An attacker can obtain data on power usage and breakers' status, and based on this information, estimate system state and determine network topology [5][6]. Then he/she can use contingency analysis methods to decide on the attack strategy (e.g., which transmission lines to open) to cause maximum damage with minimum effort. This stage can be carried out by the attacker offline to avoid possible detections.

Alternatively, an attacker can open transmission lines at random when a power system operates at high generations and load demands. Our study (see Section 4.1 for details) demonstrates that this can also easily put system into insecure states.

Attack Execution Stage. The attacker can generate legitimate but malicious commands by replaying or modifying proprietary network packets. In this paper we use the DNP3 protocol, a proprietary protocol widely used in power grid, as an example [9]. In *step 1* shown in Figure 1, a single DNP3 network packet includes four *control relay objects* to operate four breakers located in a same substation. Each control relay object uses a one-byte *device index* to indicate which breaker to operate and a one-byte *control code* to indicate the command to be performed. By modifying the *device indices* and the *control codes*, an attacker can change the selected breakers and the operations performed on them. In *step 2*, in order to hide the system changes, the attacker can intercept network packets (and/or alter the packets' payload) sent to the control center in response to the commands. If successful, the attacker can open four transmission lines simultaneously and put the system into an insecure state while providing the control center with measurement data indicating error-free operation of the substation.

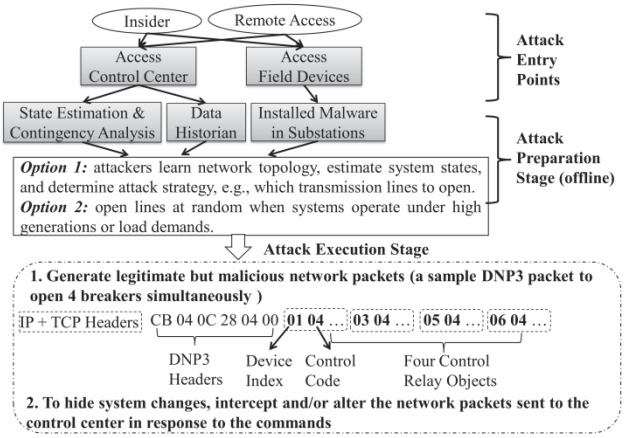

Figure 1. Attack Steps to Cause Malicious System Changes

3. Semantic Analysis Framework

As shown in Figure 2, the semantic analysis framework involves extracting control commands from SCADA network packets, obtaining measurements from sensors in substations, and triggering contingency analysis software to estimate possible consequences of executing the commands. Based on the analysis results, we generate alerts for the control center.

To collect control commands and sensor measurements, distributed IDS instances can be deployed in local area networks used in a control center and substations (e.g., *IDS instances* #1 and #2 respectively in Figure 2). IDS instances communicate with each other using standard security protocols such as SSL/TLS. We assume that the integrity of the contingency analysis software is ensured, for example, by running a redundant instance of the software in an isolated environment.

In our framework, the IDS instance associated with the control center (*IDS instance* #1) carries out the task of semantic analysis for two reasons: (i) the current contingency analysis software still requires the system's global state and significant computation capabilities, which may not be practical to provide at the level of a substation and (ii) the contingency analysis software is triggered as soon as the IDS instance #1 observes control commands issued from the control center (*activity A*). In this way, the analysis overlaps with the transmission of the commands (*activity B*); hence, the latency can be reduced.

Note that at this stage, state estimation and contingency analysis are performed by the centralized IDS (*IDS instance* #1) located at the control center. However, the cross-IDS communication (between the *IDS instance* #1 and *instance* #2) forms a trusted network environment which makes it possible to distribute the analysis into substations if more computing capabilities are deployed there in the future.

Furthermore, the centralized IDS (which integrates an analyzer for DNP3 network packets) with extended capabilities improves overall security of the SCADA system. While one may argue that the business case may still need to be made, our study indicates that the investment in building new generation IDS is worthwhile.

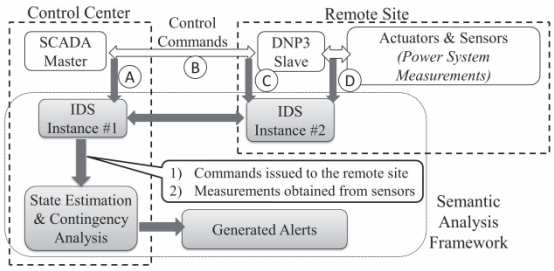

Figure 2. Semantic Analysis Framework

In order to accurately predict the state changes due to the execution of a command, two pieces of information are needed for the semantic analysis: (i) specific parameters related to the control command issued to substations at remote sites (e.g., the device indices and the control codes in the DNP3 network packets that control relays as shown in Figure 1) and (ii) trusted measurements obtained from sensors at substations.

Importantly, operations of our semantic analysis framework do not impact the normal functioning of SCADA systems, i.e., there is no additional delay introduced in the communication between the SCADA and substations.

1) Monitoring Control Commands. For each IDS instance, the Bro intrusion detection system is adapted to analyze network packets transmitted using SCADA protocols [3]. The analyzer allows the IDS to focus on *critical* SCADA commands that can operate substation devices.

One way to distinguish *critical* commands from noncritical ones is presented in Table 1. This classification is made in the context of the DNP3 protocol. The *read* commands are "passive," meaning that they do not make any changes to substations. The *write* and *execute* commands are more "invasive," and can reconfigure or change a substation state. Consequently, we regard the *write* and *execute* commands as more *critical* than the *read*. Based on this classification, IDS can select critical commands to trigger the corresponding semantic analysis. Those control functionalities are common in power grid; thus, that classification can also be found for other proprietary protocols.

2) Collecting Measurements. The semantic analysis needs the system state estimated by measurements from all substations. We use IDS instances to directly monitor and store measurements that the control center periodically collects from each substation. As sensing devices are being equipped with Ethernet interfaces, the IDS instance at the substation (*IDS instance #2* in Figure 2) collects local measurements that come directly from sensors (*activity D*). Such measurements are trusted in our threat model.

Table 1. Command Classification Based on DNP3

Command Type	Description
Read	Retrieve measurements from remote substations, e.g., read binary outputs
Write (Critical)	Configure intelligent field devices, e.g., open, edit, and close a configuration file
Execute (Critical)	Operate actuators or sensors, e.g., open or close a breaker of a relay

Some SCADA protocols can further reduce the effort required to collect measurements. For example, the DNP3 protocol allows the control center to retrieve from substations "events" that include changes in measurement data instead of the actual data, to reduce the amount of the network traffic. Also, it was recently proposed that state estimation and contingency analysis can be performed based on measurements from a specific set of substations [10], so less data need to be collected.

Unlike periodical state estimation continuously performed in the control center, the proposed semantic analysis framework estimates the system state only when a critical command is observed, to avoid unnecessary computations.

3) Checking the Integrity of Network Packets. As the control network and the field device (e.g., DNP3 slave in Figure 2) are not trusted under our threat model, the measurement data from substations can be corrupted by attackers such that the semantic analysis may produce invalid results. Similarly, a command can be modified after the semantic analysis such that the malicious command can avoid detection and still be executed at the substation.

We rely on the cross-IDS communication to check the integrity of network packets by comparing the packet payloads observed at different locations (e.g., through *activities A*, *C*, and *D* in Figure 2). The differences between the packet payloads can be used to validate the integrity of the measurements and commands that are transmitted over the vulnerable control network and the untrusted field devices. All these actions are supported by the IDS system.

4) Responses to Intrusion Detection. In this paper, we only consider the design of passive IDS which does not affect power grid operations. However, responses to intrusions play another important role, as physical damage caused by an attack can be very difficult (if not impossible) to reverse.

The proposed semantic analysis can be extended to assist in deciding on appropriate responses based on the estimated consequence of an attack. If an attack causes devastating system damage instantly, a rapid preemptive response is required. For example, deployment of intrusion prevention systems (IPSes) can provide a way to delay the command until further investigation is conducted. As indicated in [11], the maximum latency of automatic monitoring and control information from external units to substations is up to one second, which gives sufficient time to finish the semantic analysis (see Section 4.2.1). Note that the IPS can change system states and may become an attack target, therefore careful analysis in future work is required to ensure that

in strive to better protect the system, we do not introduce new vulnerabilities that can be exploited by potential attackers.

4. EVALUATION

We implemented a SCADA master and a DNP3 slave based on the open DNP3 library to produce synthetic DNP3 network traffic [9]. The traffic included DNP3 network packets, representing *read*, *write*, and *execute* commands (see Table 2). As indicated in the third column of Table 2, the generated traffic consisted of *read* commands issued periodically (every second) and used two Poisson processes with different event arrival rates to simulate the *write* and *execute* commands.

In our attack scenario, the maliciously crafted commands syntactically match the format of valid commands. Consequently, the same SCADA master is used to issue both legitimate and malicious commands.

Table 2. Control Command Simulation

Cmd Type	Description	Event Pattern
Read	Request to read (i) static data and (ii) event data from relays	Periodic event with interval of 1 second
Write	Request to (i) update the static configuration file and (ii) open/close an application in a relay	Poisson process with average command arrival interval of 50 seconds
Execute	Request to open/close a random breaker of a relay	Poisson process with average command arrival interval of 100 seconds

In order to demonstrate our approach, we use the IEEE 30-bus test system for simulation. The topology of the 30-bus test system is shown in Figure 3. We label all 30 buses, 41 transmission lines, 6 generators, and 20 load units to facilitate further discussions.

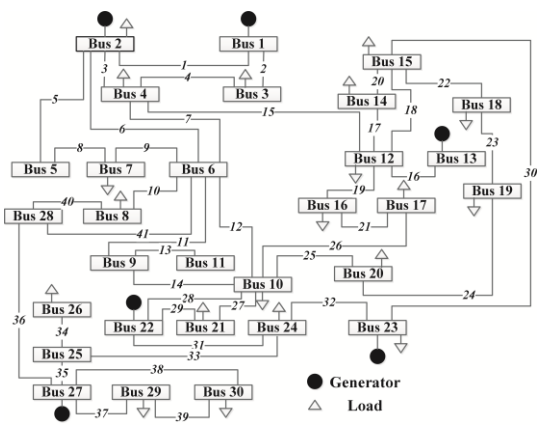

Figure 3. IEEE 30-bus System

We adopt Matpower, a Matlab toolbox for power flow analysis [12]. When the DNP3 slave receives a command, the measurements and/or the network topology are modified accordingly (see Section 4.1 for details). The state estimation module in Matpower adopts the Newton–Raphson algorithm to generate corresponding system state [5][12].

As illustrated in Figure 4, after obtaining a system state, we check the number of transmission lines in insecure conditions and use this number as an index to characterize the severity of the system changes. Those results are stored locally to validate detections made by IDS. We adopt two line-loadability criteria [13] to check line conditions: (1) the *voltage drop limit* – on two ends of a single transmission line, the voltage at the receiving end (V_R) and the voltage at the sending end (V_S) should satisfy the operational condition $V_R/V_S \geq 0.95$ and (2) the *steady-state stability limit* that decides the maximum power that a line can carry. That limit is directly related to the physical characteristics of the transmission line, e.g., capacitance, inductance, and resistance. In our experiment, we select the recommended limit values included in the Matpower source code package [12].

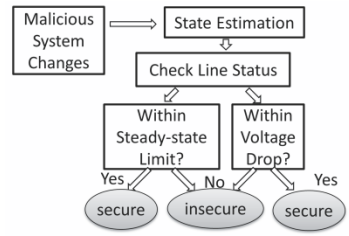

Figure 4. Procedure to Check Power System States

4.1 Case Study

In this section, we first present how malicious commands can affect a power system's steady state and then how IDS is implemented to detect the attacks.

1) Malicious System Changes. Figure 5 shows how the 30-bus system reacts to certain physical changes. The *x*-axis represents types of changes we consider: (i) increase the output of the generator attached to the bus 2, 13, 22, 23, and 27 by 50% (*Generation*), (ii) increase demands from all load units by 50% (*Load Demand*), and (iii) open 3 transmission lines at random (*Line Outage*). As there are different ways to select 3 lines out of 41 lines, we randomly select 150 cases in our experiments. We also consider making all those physical changes simultaneously (*All Changes*). Following the procedure in Figure 4, we determine the number of lines that are in insecure conditions (*y*-axis in Figure 5) based on both voltage drop (*voltage drop*) and steady-state stability limits (*stability*) and use these metrics as an indicator of the severity of the system changes.

When we launch attacks by making a single system change (the three leftmost groups of bars in Figure 5), at most 4 transmission lines are put in insecure conditions instantly if the voltage drop limit is adopted. Furthermore, when we perform attacks that make coordinated system changes (simultaneously increase generation, increase load demands, and open three lines), up to 9 additional transmission lines are put in insecure conditions. That happens when lines 15, 28, and 29 are open with the increase in generation and load demand.

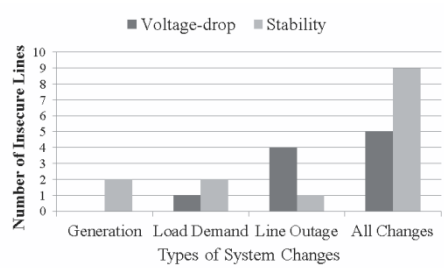

Figure 5. Effects of Physical Changes on the 30-bus System

To evaluate the possible cascading effect, we further open the 9 insecure lines and perform the power flow analysis again. The consequence is a blackout, indicated by the fact that the power flow analysis algorithm does not converge within a reasonable number of iterations (100 times for the Newton–Raphson algorithm). That means based on the new system state, a solution in which the generation (or energy supply) meets all load demands

cannot be found. Through further analysis, we find that opening the 9 insecure lines results in an isolation of a generator attached to bus 2. The remaining generators cannot meet the load demands even if all of them operate to deliver the maximum allowed power.

The study shows that the coordinated attack can put a system into more insecure states. Furthermore, intelligent attackers can avoid making changes on many physical components, which would easily draw suspicion. Instead, attackers can choose to perform attacks at a vulnerable time when the power system increases the power supply because of increased loads (e.g., because of the hot day, the power supply in the Midwest on June 28, 2012, was about 70% higher than the supply on June 2, 2012 [14]). Alternatively, attackers can target vulnerable lines, such as the lines connected to a generator, as indicated in our case study. An isolation of a generator because of power line outages can cause a power shortage or a blackout. For example, in the 30-bus test system, we can further find that in another case in which lines 1, 3, 5, and 6 are opened without any increase in the power supply (thus the generator on bus 2 is isolated), a blackout can also happen.

2) IDS Implementation. To protect against the attack scenario discussed in Figure 1, one could patch and upgrade systems to prevent attackers from accessing sensitive information on the system configuration. However, that approach is difficult to apply in the current power grid. Instead, we use the semantic analysis framework to analyze critical control commands and detect any tampering with the network packets.

We use system set up introduced in Figure 2. Based on different roles of the two IDSes, different security policies are implemented. We used Bro's scripts to implement the policy to check the integrity of network packets; Bro also supports communications between IDSes. The IDS instance in the control center performs the semantic analysis and is responsible for detecting malicious commands (issued or replayed) by the control center. To perform this task, we implemented a contingency analysis component in Matpower. The component adopts the fast decoupling algorithm to perform power flow analysis [5] and extends the algorithm with the decision procedure shown in Figure 4. The fast decoupling algorithm is commonly used for contingency analysis in today's energy management software; it achieves low detection latency at the expense of reduced computation accuracy [5]. In this work, as all attack cases put systems into insecure conditions instantly, IDSes can detect *all* of them based on this algorithm.

4.2 Performance Evaluation

As the semantic analysis runs independently without affecting power grid operations, its performance only affects the detection capability. The performance assessment of the semantic analysis is conducted on a test-bed which consists of a physical machine with an Intel i3 (3.07 GHz) quad-core and 4 GB memory, running Ubuntu 10.04 operating system.

1) Execution Time of Analyzing Critical Commands. In the first sub-section, we evaluate the execution time of the semantic analysis. For this purpose, we extend the analysis on 9-bus, 14-bus, 30-bus, 118-bus, and 300-bus IEEE test systems.

We enable the SCADA master to issue *read*, *write*, and *execute* commands to the DNP3 slave following the communication pattern described in Table 2. The SCADA master is configured to simulate 24 hours of operations. During that period, approximately 77,000 *read* commands, 1,800 *write* commands, and 900 *execute* commands are issued. We measure the average execution time of network monitoring (e.g., filtering out noncritical commands and extracting parameters of critical ones) and the triggered contingency analysis for each IEEE test system.

Figure 6 presents experimental results. The primary *y*-axis (left one) presents the time of running the contingency analysis software (*Contingency*), while the secondary *y*-axis (right one) presents the time of the network monitoring (*Network*). Both analyses are measured in milliseconds (ms).

For each IEEE test system, the same network monitoring is performed, so small variations of analysis time (*Network*) are found among different test systems. The triggered contingency analysis usually requires power flow analysis and the decision procedure illustrated in Figure 4 and, hence, more time for computations. In our experiments, approximately 70 ms and 170 ms on average are spent to perform the contingency analysis for the 9-bus and 300-bus systems, respectively.

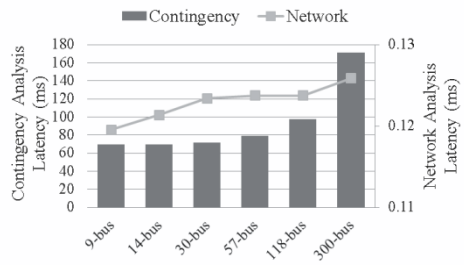

Figure 6. Execution Time Breakdown

The time spent on the contingency analysis is almost three orders of magnitude higher than the time of the network monitoring, because of the complex computations. Spending so much time on every single network packet can make the IDS miss packets if the throughput of the network traffic is large. However, the proposed semantic analysis benefits from two things. First, the throughput of network traffic involved to carry critical commands in power systems is still low. Though data are usually polled once every few seconds [10], many critical commands that are used to operate substation devices are issued manually; therefore, the intervals between control commands are on the order of minutes. Second, there is usually a limited number of types of critical commands. Thus, the IDS can ignore many uncritical commands to reduce the frequency of the semantic analysis.

2) Network Throughput. In this sub-section, we evaluate the throughput of the IDS instance equipped with the security policy to perform integrity checking on network packets. We use two throughput metrics for evaluations: the number of bits processed per second (bps) and the number of packets processed per second (pps).

We exploit the DNP3 slave to modify measurement data or commands when network packets arrive. Then we generate the same synthetic network traffic as in Section 4.2.1 and collect a 1 GB packet trace. The whole trace includes a total of 2,040,000 DNP3 packets. The DNP3 analyzer integrated in the IDS processes the packet trace offline. We performed 10 experimental runs to measure the average execution time.

Table 3. Throughputs of IDS to Check Network Integrity

Metrics	Throughput
Packets per second	11,114
Megabits per second	43

Based on the evaluation (for which results are shown in Table 3), more than 11,114 DNP3 network packets are processed every second. In the power grid environment, the control center usually polls measurements once every few seconds [10]. Based on those figures, we anticipate that the proposed DNP3 analyzer can monitor network packets of power systems consisting of more than 10,000 devices. To monitor a larger scale of power system, IDS instances can be connected in a hierarchical structure.

Notably, validating the integrity of network packets requires communications between IDS instances. In the context of the DNP3 protocol, a network packet with the length of about 1K bytes can carry more than 1000 binary or 250 32-bit floating point measurements. Thus, typical SCADA networks seldom use bandwidth bigger than 8 Mbps (Megabits per second). As a result, delivering that network traffic for IDSes will not impair IP-based network infrastructure with 100-Mbps links or even 1-Gbps links that are commonly deployed.

5. RELATED WORK

How power systems react to physical changes, such as line outages, has been studied [2][15]. In our attack scenario, we further associate the physical changes with the cyber vulnerabilities. Such physical changes, if initiated by legal but malicious commands, can easily bypass previously proposed network IDSes that rely on deviations from a predefined or normal communication pattern [3].

[16] proposes to include physical information to detect attacks, a concept that is similar to our semantic analysis. However, the proposed semantic analysis framework has two important differences. First, [16] relies on single-packet attack signatures to analyze network traffic, while our work can fully extract and analyze all SCADA-specific semantics. Consequently, we can provide much better accuracy and flexibility to decide when and how to use the physical knowledge. Second, [16] relies on a centralized image to detect attacks, while we deploy distributed IDS instances to further detect the compromise of measurements or control commands during the communication.

6. CONCLUSIONS

In this paper, we rely on distributed IDS instances to perform semantic analysis on SCADA network packets. Selected network IDS instances leverage the existing contingency analysis software to estimate the execution consequences of the control commands transmitted over the vulnerable SCADA network. Distributed IDS instances establish trusted communication to detect compromises of measurements or control commands. Thus, the semantic analysis can provide trusted detections on malicious control commands.

We evaluated the proposed semantic analysis framework on the IEEE 30-bus system. Based on this system, we study the effects of malicious control commands that can be achieved through exploitation of certain system vulnerabilities. With sufficient system knowledge, an attacker can put the system into an insecure state with a maliciously crafted control command. To evaluate the performance of the proposed semantic analysis, we measure the time to perform the network monitoring and the triggered contingency analysis. Based on the results, we find that the proposed semantic analysis framework shows promise for providing efficient detection in today's power grid.

In future work, we will focus on preemptive analysis on both cyber and physical knowledge from power grid. Consequently, appropriate mechanisms for responding to malicious control commands can be designed.

7. ACKNOWLEDGMENTS

This material is based upon work supported in part by the Department of Energy under Award Number DE-OE0000097, by the National Science Foundation under Grant No. OCI-1032889, and by Infosys Limited. The authors would like to thank Prosper Panumpabi from University of Illinois at Urbana-Champaign for his insightful discussions.

8. REFERENCES

[1] Electrical grid in U.S. penetrated by spies. *The Wall Street Journal*, p. A1, April 8, 2009.

[2] Glover, J. D., Sarma, M.S. and Overbye, T. 2011. *Power System Analysis and Design*, 5th ed., Cengage Learning.

[3] Lin, H., Slagell, A., Di Martino, C., Kalbarczyk, Z. and Iyer, R.K. Adapting Bro into SCADA: Building a specification-based intrusion detection system for the DNP3 protocol. In *Proc. of 8th Annual Cyber Security and Information Intelligence Research Workshop,* 2013).

[4] Falliere, N., Murchu, L. and Chien, E. *W32.Stuxnet dossier.* Symantec Security Response, 2011.

[5] Monticelli, A. Electric power system state estimation. In *Proceedings of the IEEE*, Vol.88(2), 2000.

[6] Prais, M. and Bose, A. A topology processor that tracks network modifications. 1998. *IEEE Transactions on Power Systems* (August 1988), vol. 3, no.3, pp. 992-998.

[7] Liu, Y., Ning, P., and Reiter M. False data injection attacks against state estimation in electric power grids. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS'09, 2009.

[8] Bobba, R., Rogers, K., Wang, Q., Khurana, H., Nahrstedt, K., and Overbye T. Detecting false data injection attacks on DC state estimation. In *Preprints of the First Workshop on Secure Control Systems,* SCS 2010.

[9] Open DNP3 Group. 2012. *DNP3 – Distributed Network Protocol 3.0–Google project hosting*. Online. Available: http://code.google.com/p/dnp3/.

[10] Yang, T., Sun, H., and Bose, A. Transition to a two-level linear state estimator – Part 1: architecture. 2011. *IEEE Transactions on Power Systems*, 26(1) 2011).

[11] IEEE standard communication delivery time performance requirements for electric power sub-station automation. IEEE Std 1646-2004, 2005.

[12] Zimmerman, R. D., Murillo-Sánchez, C. E., and Thomas, R. J. MATPOWER: Steady-state operations, planning and analysis tools for power systems research and education. 2011. *IEEE Transactions on Power Systems*, 26(1), 2011).

[13] Gutman, R., Marchenko, P., and Dunlop, R. Analytical development of loadability characteristics for EHV and UHV transmission lines. *IEEE Transactions on Power Apparatus and Systems*, PAS-98(2), 1979.

[14] Midwest Independent Transmission System Operator, Inc. 2012. *June 2012 Monthly Market Assessment Report*.

[15] Lesieutre, B., Pinar, A., and Roy S. Power system extreme event detection: The vulnerability frontier. In *Proceedings of 41st Annual Hawaii International Conference on System Sciences* (January 2008).

[16] Carcano, A., Fovino, I., Masera, M., and Trombetta, A. State-based network intrusion detection systems for SCADA protocols: a proof of concept. *Critical Information Infrastructures Security, Lecture Notes in Computer Science*, vol. 6027, 2010, pp. 138-150