Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

MELODY: SYNTHESIZED DATASETS FOR EVALUATING INTRUSION DETECTION SYSTEMS FOR THE SMART GRID

Vignesh Babu
Rakesh Kumar
Hoang Hai Nguyen
David M. Nicol
Kartik Palani
Elizabeth Reed

Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

ABSTRACT

As smart grid systems become increasingly reliant on networks of control devices, attacks on their inherent security vulnerabilities could lead to catastrophic system failures. Network Intrusion Detection Systems(NIDS) detect such attacks by learning traffic patterns and finding anomalies in them. However, availability of data for robust training and evaluation of NIDS is rare due to associated operational and security risks of sharing such data. Consequently, we present Melody, a scalable framework for synthesizing such datasets. Melody models both, the cyber and physical components of the smart grid by integrating a simulated physical network with an emulated cyber network while using virtual time for high temporal fidelity. We present a systematic approach to generate traffic representing multi-stage attacks, where each stage is either emulated or recreated with a mechanism to replay arbitrary packet traces. We describe and evaluate the suitability of Melodys datasets for intrusion detection, by analyzing the extent to which temporal accuracy of pertinent features is maintained.

1 INTRODUCTION

The smart grid is representative of a cyber-physical system, which uses a networked set of devices that sense its state and take appropriate control decisions (e.g. open/close a circuit breaker). Due to vulnerabilities in the communication protocols, end-host firmwares and control algorithms, the smart grid's control network becomes a potential attack vector. The recent trend in attacks on power grids indicates usage of sophisticated attack campaigns characterized by multi-stage exploits (Falliere, Murchu, and Chien 2011), (Bencsáth, Pék, Buttyán, and Felegyhazi 2012), (Assante and LEE 2015). In a typical attack campaign, the attacker creeps through different network layers by stealing legitimate credentials and/or exploiting vulnerabilities in the network services, progressively acquires more privileged access to one or more of the "critical assets" before finally delivering the attack, e.g. opening multiple circuit breakers at once (Lee, Assante, and Conway 2016). Such multi-stage attacks are observable on both the cyber (e.g. packet counts measured at network devices) and physical (e.g. power values measured by a phasor measurement unit) attributes of the smart grid system.

Machine learning based network intrusion detection systems (NIDS) can detect such multi-stage attacks by observing statistical patterns in such attributes. These systems are trained with historical data comprising of normal *background* and attack traffic; two kinds of training occurs, one on normal traffic, so as to be able to detect abnormalities by deviation from the norm, and separately on specific patterns from known attacks, to detect specific abnormalities. We are concerned with both kinds of training. The accuracy of a

machine learning based NIDS depends on the quality of the training datasets. Specifically, the datasets must capture the heterogeneity observed in the topologies and communication protocols of production smart grid networks (Sommer and Paxson 2010). However, such datasets are difficult to acquire due to the associated security risks and operational resource constraints (Pang, Allman, Paxson, and Lee 2006)(Sommestad and Hallberg 2012); operators whose system produce data needed for training are reluctant to share it broadly. This is problematic for researchers in machine-learning based of NIDS, minimally, to be able to identify features in normal traffic which strongly differentiate normal from abnormal behavior. This difficulty motivates the use of synthetic datasets for training and evaluation of NIDS. However, previous efforts to synthesize such datasets have been criticized for irregularities in the generated packet headers (Mahoney and Chan 2003). The criticism has also highlighted the inaccuracies in modeling the background traffic patterns in the networks that are connected to the Internet (McHugh 2000)(Sommer and Paxson 2010). However, the traffic flows on the isolated smart grid networks tend to be regular (Barbosa, Sadre, and Pras 2016), thus the background traffic for them can be potentially modeled. Finally, while there are adversarial traffic traces available in the public domain (Sommestad and Hallberg 2012), they are not immediately useful to train an NIDS for a given smart grid network.

To that end, we developed Melody, a scalable data generation framework that integrates a communication network emulation with an industry standard power system simulation. The fundamental goal of Melody is to synthesize high temporal fidelity datasets that capture the interactions between the cyber and physical elements of a smart grid system during normal operating conditions, to train on normal behavior, and when the system is under attack, to training on certain types of abnormal behavior. The fidelity is achieved in part by using a virtual time system (Gupta, Yocum, McNett, Snoeren, Vahdat, and Voelker 2005) to allow fine-grained control over the execution of each system process that participates in the dataset synthesis. Melody avoids packet header irregularities by generating packets either by emulating actual production software or in case such emulation is not possible, by embedding packet traces collected from arbitrary networks in the modelled network. Its input constitutes of the configuration of the communication network (e.g. topology, link latencies, routing state) and specifications of the background traffic and multi-stage cyber-attacks. Its output is a dataset that constitutes of packet capture files obtained by capturing network packets at chosen communication links in the network over a specified period of time. Our major contributions include:

- A framework to synthesize datasets that represent a variety of traffic backgrounds in a smart grid network, thereby allowing training and evaluation of an NIDS from a representative sample of production scenarios.
- A mechanism to generate traffic representing multi-stage attacks launched by advanced and persistent adversaries. Melody synthesizes datasets such that the adversarial traffic is embedded within the background traffic by using a combination of emulation and traffic replay mechanisms.

Our evaluation shows that the datasets synthesized by Melody have same temporal characteristics as the data obtained from a real network. Furthermore, Melody generates datasets representing networks that contain thousands of simultaneous flows while maintaining temporal and causal fidelity.

The rest of the paper is organized as follows: Section 2 introduces background, Section 3 describes Melody's architecture and its integration with a virtual time system, Section 4 describes the traffic synthesis mechanisms Section 5 describes how attack traffic is synthesized, Section 6 presents an evaluation. Section 7 is a case study of how Melody is used to synthesize traffic for NIDS research, Section 8 presents related work and Section 9 concludes.

2 BACKGROUND

2.1 System Overview

Smart grid communication networks use a two layered architecture comprised of a corporate network and a fieldbus/control network. The corporate network facilitates IT management, operator control and storage

& analysis of process control data. The control network connects controllers and field devices through a resilient topology comprised of multiple switches. It is typically interfaced with the corporate network through firewalls and DMZs.

2.2 Attack Model

Recently successful sophisticated attack campaigns on Industrial Control Systems (ICS) like Stuxnet (Falliere, Murchu, and Chien 2011) , Flame (Bencsáth, Pék, Buttyán, and Felegyhazi 2012), Havex (Assante and LEE 2015) and the Ukrainian power grid attack (Lee, Assante, and Conway 2016) launched multi-stage attacks targeting certain attributes/devices of a specific Industrial control system (ICS). Such advanced persistent threats are known to deploy malware which interact with an outside server and execute in stages by first replicating inside the corporate network before spreading to the fieldbus network. Further, such attack campaigns typically remain dormant for long periods of time before the final attack stage is delivered.

We consider an attacker who devises a multi-staged attack, such conceptualized in (Assante and LEE 2015) as the *ICS cyber kill chain*. The attacker can legitimately access the smart grid cyber network (e.g. using stolen VPN credentials), as has been observed in the field. He may then download malware, discover the network structure and start exploiting multiple vulnerabilities to ultimately be able to directly interact with the smart grid's control elements. This sequence leaves evidences of an ongoing attack in three different data sources, namely network traffic, power measurements, and control commands. By aiming at reproducing the evidences of attack from all three data sources, Melody encourages the development of anomaly detection schemes that look for correlation of events in multiple time-series to detect anomalous activities.

2.3 Virtual Time Systems

Melody must be able to emulate large smart grid networks, using limited CPU resources relative to the number of devices in the emulation. To capture temporal fidelity virtual time systems alter and control processes perception of time. Each emulated process is assigned its own virtual clock. Each virtual clock can advance faster or slower than real time and the rate of advancement of the virtual clock w.r.t the actual time is also referred to as the Time Dilation Factor (TDF). In this paper, we use TimeKeeper (Lamps, Nicol, and Caesar 2014), a linux based virtual time system, which embeds selected processes in virtual time. TimeKeeper intercepts and modifies all time system calls offered by the operating system. It maintains temporal synchrony among all processes under its control by precisely controlling each process's execution time. Our previous works (Babu and Nicol 2016)(Lamps, Adam, Nicol, and Caesar 2015), demonstrated emulation of PLC Networks in virtual time and integration of TimeKeeper with popular network emulators and simulators CORE, ns-3, S3FNet. We supported the work reported here with an additional extension to mininet. (Mininet 2016) .

3 ARCHITECTURE

Melody uses PowerWorld Simulator (Power World 2016) to simulate electrical behavior of the power grid. Melody uses a plugin for PowerWorld which allows an external process to control the state of the entities (e.g buses, generators) in the power simulation. However, PowerWorld does not explicitly model the grid's cyber aspects. We therefore coordinate a separate network emulation with PowerWorld (Figure 2). The overlaid communication network is emulated using mininet (Mininet 2016). The capability of modifying the mininet topology and the link delay configurations allows us to model a variety of networks.

A proxy process serves as an interface between the power simulator and the network emulator. Control commands from an emulated control node (e.g. a SCADA master; the red node in Figure 2) are routed through the emulated network to the destination host (e.g. an RTU that controls a circuit breaker) and later transferred from the host to the power simulator through the proxy. Responses from the power simulator (e.g. voltage magnitude and angle measurements) are re-routed back to the control station (Figure 1).

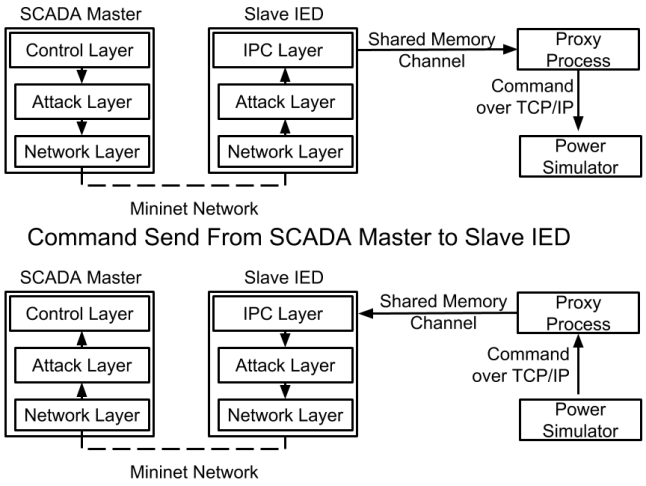

Figure 1: Integration between cyber and physical components via Proxy.

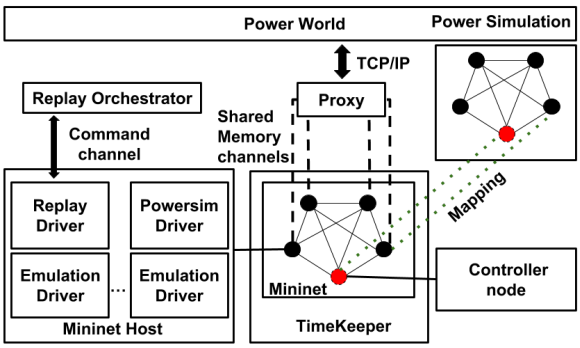

Figure 2: Melody Architecture.

Each emulated host can run three types of driver processes: emulation driver, replay driver, and powersim driver. Emulation drivers produce traffic in the testbed by spawning processes that interact with each other. A replay driver initiates traffic replay actions at specified times (Section 4). A powersim driver is also started in each node to convey PowerWorld grid state data across the cyber network to and from the proxy process. The powersim driver spawns three threads: an application layer thread (IPC layer) to emulate smart grid applications, an attack layer thread which can intercept, modify and inject application level packets, and a network layer thread for handling packet transmissions and receptions (Figure 2).

3.1 Virtual Time Integration

We introduced virtual time into Melody through TimeKeeper (Lamps, Nicol, and Caesar 2014). TimeKeeper provides an API to add processes to its control and it can automatically detect and control other processes that may be spawned by the current set of controlled processes. At the beginning of emulation, a script starts all drivers and instructs TimeKeeper to control them. The emulation proceeds in rounds where the virtual time of each controlled process increases by a "time slice" each round. The time slice is usually set as the greatest common divisor of all link latencies in the network to avoid causal violations. At the end of each round, the startup script instructs PowerWorld to advance its simulation time by the round duration. PowerWorld uses any input packets (i.e. from the proxy) gathered during the last round and responds with output packets (i.e. to the proxy) which are delivered to the appropriate nodes at the start of the next round.

Tracking virtual time requires additional Linux kernel modifications and extensions to TimeKeeper. Mininet uses a Linux kernel module called netem to enforce propagation delays and bandwidth restrictions on links in the topology. Transmitted packets are rate limited and queued for a period equal to the specified link latency before being sent from the interface. The kernel modifications enforce the queuing delay in virtual time with high accuracy. Furthermore, we enforce virtual timestamps on packets transmitted from an interface to ensure that packets are recorded in virtual time by packet capturing tools such as tcpdump and Wireshark. Due to space constraints, we do not delve further into the kernel modification details here.

4 TRAFFIC GENERATION

When the smart grid is not under attack, the traffic observed at all links are called *regular traffic*. When the grid is under attack, some links will additionally carry traffic generated by the attacker, or *attack traffic*. Both regular and attack traffic might contain benign and suspicious network activities. For example, a

single successful VPN connection is a benign event but might belong to the attack traffic if the attacker has obtained a legitimate login credential. Similarly, a staff member can repeatedly mistype his/hers VPN login password, causing multiple failed login attempts, but this suspicious event belongs with regular traffic. The attack traffic usually occurs at different links during multiple phases of an attack campaign. Consequently, selecting the collection of links to tap is an important aspect of attack detection as it determines whether an NIDS has full, partial, or even zero visibility of an ongoing attack. We generate the regular and attack traffic using two techniques, *traffic emulation* and *traffic replay*. A combination of these techniques allow us to recreate a variety of network level conditions that may occur in a smart grid network.

4.1 Traffic Emulation

Melody allows emulation of traffic with actual processes (e.g. sshd, ping, httpd, and so on) spawned on mininet hosts by emulation drivers. This capability enables emulation of programs with known vulnerabilities and subsequent capture of network-level behavior of any attempted attacks. Emulation drivers can spawn processes at precise offsets of time after the start of the experiment, once or multiple times during the experiment. Users can specify traffic emulation patterns which define the time instants at which emulated processes are spawned. We allow three types of traffic emulation patterns:

- *one-shot* patterns where the process is spawned only once. It can be used to emulate rarely-occurring one time events like uploading device firmwares.
- *periodic* patterns where the process is spawned periodically. It can be used to emulate sensor measurements and control commands.
- *random* patterns where the process spawn instants are drawn from probability distributions. It can be used to emulate frequently occurring human-initiated activities like VPN and remote ssh logins.

4.2 Traffic Replay

Melody's replay mechanism enables traffic synthesis for services and attacks that are otherwise difficult to emulate inside mininet hosts due to operational constraints or difficulties exploiting vulnerabilities that only exist in an actual network. The input to the replay mechanism is a recorded trace of network traffic between a pair of hosts collected from a tap point along the path between the hosts in any network. The trace is in PCAP format (libpcap Developers 2016) which contains a sequence of packets with capture timestamps. Before the replay starts, the source and destination IP addresses of the packets are modified to match the corresponding hosts' IPs in mininet. The mechanism ensures the following two properties in the replayed trace:

- Causality: The order of packets in the replayed trace is same as in the original trace
- Timing Fidelity: The time between a received and sent packet by a given host in the replayed trace (e.g. processing delay by a webserver) is same as the original trace.

The replay mechanism is two-tiered: The first tier involves a process called replay orchestrator (RO), which synchronizes the actions of the second-tier host replay drivers (Figure 2). These actions include pre-loading or sending the packets in an input PCAP file. During pre-loading, the driver on each host scans the PCAP and compiles a list of packets to be sent. For each packet, the driver builds a list of packets received by the host before the packet is sent, and calculates the capture timestamp difference Δt between that packet and the last received packet. It uses Δt to compute and store a processing delay estimate for each packet, by subtracting the round-trip time (RTT) estimate between the source and destination hosts in the original trace from Δt. If the original PCAP contains a TCP flow, then RTT can be estimated from the minimum difference between the timestamps of any sent packet and its corresponding ACK packet. Same can be done for general protocol conversations (e.g. UDP), however the RTT estimate is coarser.

Finally, when the start of replay is triggered by the RO, before sending the next packet, the driver waits for all the corresponding packets it needs to receive to ensure causality. Furthermore, the driver also sleeps for the processing delay for each packet to send, before sending it, thus ensuring timing fidelity.

5 ATTACK SYNTHESIS

We model an attack with an *attack plan*, which is a step-by-step plan of actions an attacker devises in order to achieve his goal. An attack plan starts with explicit assumptions about the attacker, i.e. what machines he has compromised, what kind of attacks he is capable of performing, and what ultimate goal is he trying to achieve. Each action in the attack plan, called an *atomic attack*, allows the attacker to gain information about the network, to acquire more privileges on a compromised machine or access to a new machine, or to cause some disturbance/damage to the system under attack. At the end of the attack plan, the attacker achieves his goal, for example he successfully shutdowns several generators, leading to a system-wide blackout. In our framework, attack synthesis (i) constructs a logical and credible attack plan and (ii) executes the attack plan in the testbed so that traces can be collected at different access points.

5.1 Properties of an Attack Plan

In order to credibly represent actual cyber-attacks, attack plans must have certain properties. First, every atomic attack must abide by the underlying topology, configurations, services, and their vulnerabilities as specified in the smart grid network. For example, if no path from host A to host B exists due to a firewall rule, then there can be no atomic attack that compromises host B from host A. Second, as one atomic attack usually enables another, they must appear in the proper order in the attack plan. For example, starting with the assumption that host A is not compromised, an atomic attack that allows the attacker to perform a network scan using host A must be preceded by another in which host A is attacked and compromised. Finally, a *credible* attack plan that models an advanced persistent threat to the power grid networks should be reasonably complex in the number of atomic attacks required and vulnerabilities exploited (including zero-day vulnerabilities), among others.

5.2 Synthesizing an Attack Plan

Melody reads an attack plan in the form of an input JSON file and synthesizes each atomic attack in the specified order to generate the attack traffic at the appropriate network interfaces after emulating link delays. For the sake of modularity, each atomic attack is implemented as a separate `Python` function. Note that not every atomic attack results in discernible attack traffic (e.g. local privilege escalation). Among the ones that do, some are generated using the traffic emulation and others using the traffic embedded replay (Section 4). Table 2 shows an example of a manually constructed attack plan with respect to the network in Figure 6. In this example, we assume the attacker has already obtained a legitimate VPN account to access the power grid network. Furthermore, he knows how to perform a network scan using `nmap` and is capable of launching a password fuzzing attack.

6 EVALUATION

In this section, we present our evaluation of the mechanisms described above as the scale of entities and synthesized flows increases. Specifically, we evaluate the temporal fidelity of the datasets synthesized by Melody by using the emulation and traffic replay mechanisms. The temporal fidelity is important for the training of the NIDS because it underlies many machine learning feature selection strategies that measure some type of rate. For example, an IDS may observe the per-second packet counts for control network traffic transported via a link. If it detects any anomalies in the per-second packet counts, then the application may raise an alarm for a potential intrusion. If the mean and standard deviation of these counts are not reflected correctly in the training data, then the trained NIDS would be not be accurate when put in production.

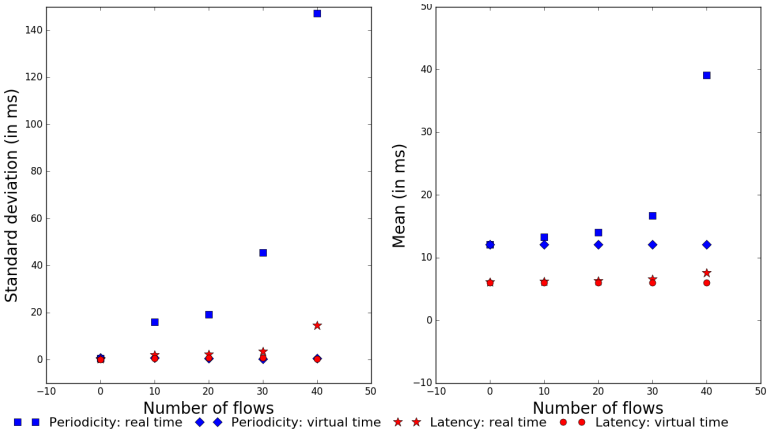

Figure 3: The observed standard deviation and means of the periodicity and latency with and without TimeKeeper. The expected mean latency is 6 ms and periodicity is 12 ms.

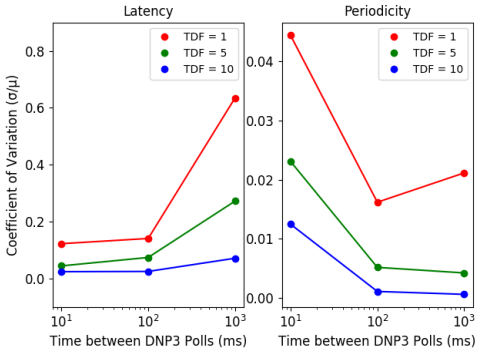

Figure 4: The coefficients of variation of the latency and periodicity of sample DNP3 traffic in the presence of 1765 total flows.

For our experiments, we used a Xeon server with 56 cores clocked at 2.0 GHz with 64 GB of RAM running Ubuntu 14.04 with a modified kernel version 3.13 to support TimeKeeper. We focus our evaluation on the fidelity of a widely used control protocol called DNP3 protocol (IEEE 2012). DNP3 specifies a designated master node and a collection of slave nodes. The master node sends a polling request to the slave nodes at a fixed polling interval and the slave nodes respond to the polling request instantaneously. The experiments described in this section measure latency and periodity of the DNP3 polling requests. In a given PCAP file, latency is measured as the difference between timestamps of a polling request and the corresponding reply packet and periodicity is the difference between the timestamps of two successive polling requests.

6.1 Evaluating Traffic Emulation

The first experiment on traffic emulation motivates the need for virtual time systems by providing a comparison of traffic synthesized by Melody with and without TimeKeeper. The second experiment demonstrates the ability of the system to generate high fidelity data at scale.

6.1.1 Virtual Time Integration

We generated PCAPs with Melody using the topology shown in Figure 8 with one host per switch and a link latency of 1 ms. The host at switch s_1 is the DNP3 master and the host at switch s_3 is a DNP3 slave. The host at switch s_1 also runs a UDP server. Every other host is running an emulated process that acts as a UDP client and also performs some compute intensive operations. This models a smart grid control network which has a single SCADA master and a few SCADA slaves which not only communicate with the master and other entities of the control center but also run some computation before sending data.

Figure 3 shows the mean and standard deviation of the DNP3 transaction latency and polling periodicity of the packets captured on link $s_1 - s_3$ for operation of Melody under two configurations: with and without TimeKeeper. The results indicate that when Melody is used without TimeKeeper and is tasked to synthesize a large number of background flows, both latency and periodicity experience higher mean and standard deviation. This is because the processes are no longer scheduled as they should. However, when TimeKeeper is used, the increased number of flows does not affect temporal fidelity of the synthesized datasets.

Interval	10	100	1000
$\mu_{periodicity}$	10.24	100.26	100.31
$\sigma_{periodicity}$	0.031	0.034	0.015
μ_{rate}	13518.97	1426.21	139
σ_{rate}	33.32	26.49	0

Table 1: Periodicity (ms) and Data Rates (Bytes/s) in the original DNP3 PCAP files with the configured polling interval (ms) in DNP3 master/slave processes.

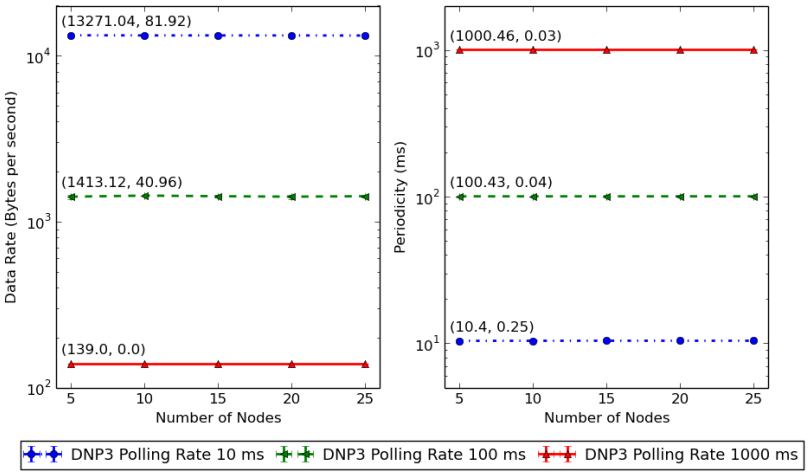

Figure 5: The observed data rates and periodicity in the replayed DNP3 PCAP files.

6.1.2 Evaluating Scalability

The topology was scaled up to a 10 switch ring with 4 hosts per switch, resulting in 1765 flows. The latency and periodicity of DNP3 traffic was measured for transmissions between a DNP3 master and slave on link $s_1 - s_3$, given different time dilation factors (TDF) and DNP3 polling rates. The coefficients of variation for the latency and periodicity shown in Figure 4 indicate that increasing the TDF reduced the variation of both measurements, allowing for greater fidelity of the emulated traffic.

6.2 Evaluating the Traffic Replay

To evaluate the accuracy of the traffic replay mechanism, we first collected a PCAP between two real hosts connected to a single physical switch. A DNP3 master on one host polled a DNP3 slave on the second host with polling intervals of 10 ms, 100 ms and 1000 ms. The PCAP files were generated at the master host and the mean round trip time between the hosts was estimated as 320 μs with a small standard deviation.

The emulated topology used in this evaluation also consisted of a single switch with a configurable number of hosts attached to it. One pair of the emulated hosts replayed the DNP3 the PCAP files, while the other hosts ran compute intensive tasks. We measured the periodicity and the data rate of the traffic between the host pair in the replayed PCAP files at a TDF of 5 and compared them with values in the original PCAP files which are shown in Table 1. Figure 5 presents the results gathered from this experiment, illustrating that regardless of the number of hosts and emulated processes in the experiment, our testbed still manages to closely maintain the same periodicity and poll rate as observed in the original PCAP.

7 CASE SUDY: DATASETS FOR NIDS research

NIDS research focuses on building monitors that can either identify malicious attacks based on known attack signatures or deviations from regular network traffic. NIDS are classified as either *signature-based*, where the IDS is given a precise description of an attack to check for, or *anomaly detection-based*, where the IDS builds a model of benign normal activity and flags any anomalous behavior as an attack. For any NIDS, exhaustive training datasets are critical. However, the lack of publicly available data is a major hurdle in the design and commercial deployment of IDSes (Sommer and Paxson 2010), (Tavallaee, Stakhanova, and Ghorbani 2010), (Shiravi, Shiravi, Tavallaee, and Ghorbani 2012). Due to the risk to critical infrastructures such as the smart grid, the organizational and legal barriers make it harder for researchers to get such datasets.

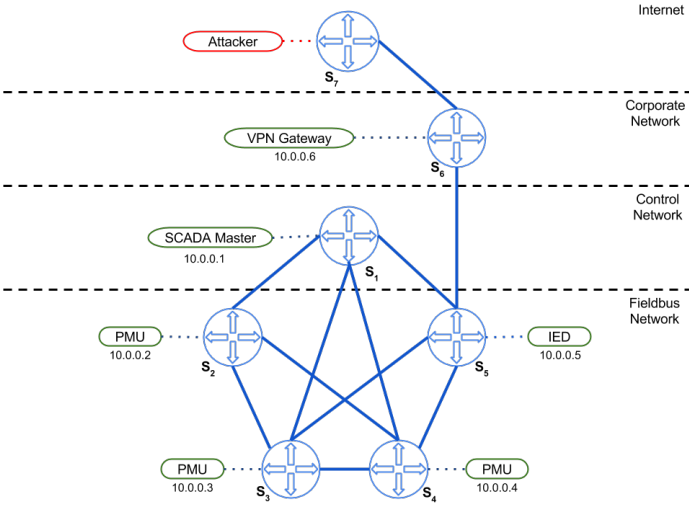

Figure 6: The cyber network topology used in the case study.

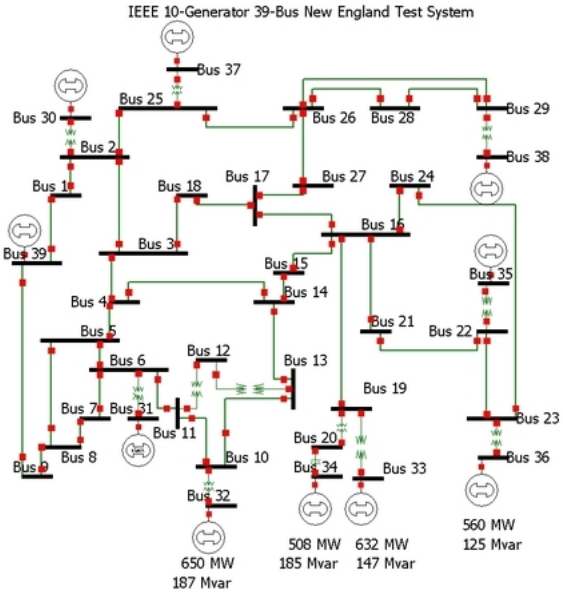

Figure 7: The power simulation topology used in the case study.

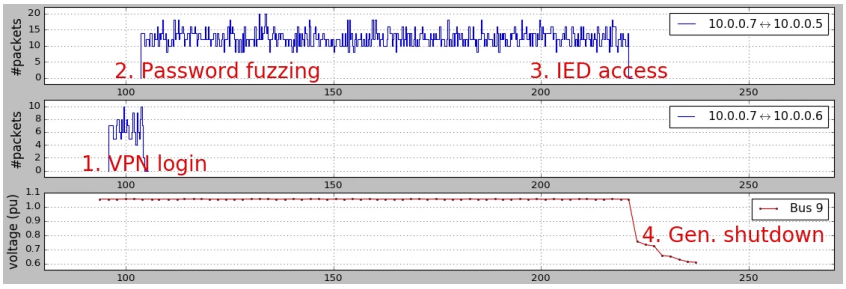

Figure 8: Demonstration of the execution of a multistage attack on the testbed.

In order to demonstrate Melody's capability to produce representative datasets, we pick cyber and power topologies and study the time series of the traffic generated by them. The network topology is shown in Figure 6 and the power topology used is a IEEE 39-bus system, shown in Figure 8. Each switch in the power topology has one connected host. Switch s_1, at the control center, has the SCADA master connected to it which acts as the DNP3 master and has a few outstation devices connected to it, which are a part of the fieldbus network. The attacker enters the network from the Internet and connects to the VPN gateway at switch s_6. The attack plan shown in Table 2 was executed to introduce an attack into the dataset.

8 RELATED WORK

Previous work has proposed the design of real testbeds for data collection from prototypical critical infrastructure control systems (Giani, Karsai, Roosta, Shah, Sinopoli, and Wiley 2008) (Morris, Srivastava, Reaves, Pavurapu, Abdelwahed, Vaughn, McGrew, and Dandass 2009) (Sommestad and Hallberg 2012). Such testbed implementations are generally expensive to build and less flexible. Power system simulation tools like PowerWorld (Power World 2016), RSCAD and RTDS (RTDS 2016) allow the modeller to study variations in the power state of the system over time but do not support incorporation of a cyber network and control model. To circumvent this problem, virtual testbeds which combine power system and network simulation like (Hannon, Yan, and Jin 2016), (Chen, Butler-Purry, Goulart, and Kundur 2014), (Saran,

Step	Atomic attack description	Execution
1	VPN Connection to the power grid network from host A at 10.0.0.7	replayed
2	Scan the power grid local network from host A using nmap	emulated
3	Launch a fuzzing attack against the IED at 10.0.0.5 from host A	replayed
4	Connect to the IED from host A via telnet	replayed
5	Perform privilege escalation on the IED to obtain administrator access	ignored
6	Reconfigure the IED to turn off two generators	emulated

Table 2: A 6-step attack plan to shutdown two generators in the power grid.

Network Topology	Power Topology
S_1	Controller (SCADA master)
S_2	Buses 2, 25, 29
S_3	Buses 22, 23, 19
S_4	Buses 20, 10, 6, 9
S_5	Generator Buses 30-39

Figure 9: The mappings between the power topology and the network topology.

Palla, Srivastava, and Schulz 2008) have been proposed. However, while they offer greater flexibility in manipulating experimental conditions to perform repeatable evaluations, these hybrid emulation-simulation systems suffer from scalability and fidelity issues. Finally, the use of time dilation in virtual time system (Gupta, Yocum, McNett, Snoeren, Vahdat, and Voelker 2005) to construct testbeds for specific applications is not novel. SVEET! (Erazo, Li, and Liu 2009) and Diecast (Gupta, Vishwanath, McNett, Vahdat, Yocum, Snoeren, and Voelker 2011) are testbeds using virtual time, however, neither is geared towards generation of network traces for NIDS training.

9 CONCLUSION

The modularity, scalability and flexibility of Melody makes it an effective tool to generate datasets that capture the diversity of communication networks in power grids. It has both academic and non-academic use: The NIDS research community can leverage it to build a database of synthetic datasets. The database could contain datasets for a variety of configurations (i.e. different power and cyber topologies, traffic backgrounds with and without multi-stage cyber-attacks). This database can be published in the public domain and used to evaluate the accuracy of a new approach to building NIDS. Similarly, Melody can also be used by the network administrators seeking to augment an existing NIDS without sharing the details of their networks to outsiders. They can use Melody to model their power system communication network with its specific configuration and generate attack datasets for potential vulnerabilities using the model instead of perturbing the actual system.

ACKNOWLEDGEMENT

This work was supported in part by the Siebel Energy Institute, and in part by by the Department of Energy under Award Number DE-OE0000097. Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

REFERENCES

Assante, M. J., and R. LEE. 2015. "The Industrial Control System Cyber Kill Chain". *SANS Institute InfoSec Reading Room*.

Babu, V., and D. M. Nicol. 2016. "Emulation/Simulation of PLC Networks with the S3F Network Simulator". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. Huschka, S. Chick, J. Jimenez, P. Frazier, T. Roeder, R. Szechtman, and E. Zhou, 1475–1486. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Barbosa, R. R. R., R. Sadre, and A. Pras. 2016. "Exploiting Traffic Periodicity in Industrial Control Networks". *International Journal of Critical Infrastructure Protection* 13:52–62.

Bencsáth, B., G. Pék, L. Buttyán, and M. Felegyhazi. 2012. "The Cousins of Stuxnet: Duqu, Flame, and Gauss". *Future Internet* 4 (4): 971–1003.

Chen, B., K. L. Butler-Purry, A. Goulart, and D. Kundur. 2014. "Implementing a Real-time Cyber-physical System Testbed in RTDS and OPNET". In *North American Power Symposium (NAPS), 2014*, 1–6. IEEE.

Erazo, M. A., Y. Li, and J. Liu. 2009. "SVEET! a scalable Virtualized Evaluation Environment for TCP". In *Testbeds and Research Infrastructures for the Development of Networks & Communities and Workshops, 2009. TridentCom 2009. 5th International Conference on*, 1–10. IEEE.

Falliere, N., L. O. Murchu, and E. Chien. 2011. "W32. stuxnet dossier". *White paper, Symantec Corp., Security Response* 5:6.

Giani, A., G. Karsai, T. Roosta, A. Shah, B. Sinopoli, and J. Wiley. 2008. "A Testbed for Secure and Robust SCADA Systems". *ACM SIGBED Review* 5 (2): 4.

Gupta, D., K. V. Vishwanath, M. McNett, A. Vahdat, K. Yocum, A. Snoeren, and G. M. Voelker. 2011. "DieCast: Testing Distributed Systems with an accurate scale model". *ACM Transactions on Computer Systems (TOCS)* 29 (2): 4.

Gupta, D., K. Yocum, M. McNett, A. C. Snoeren, A. Vahdat, and G. M. Voelker. 2005. "To Infinity and Beyond: Time Warped Network Emulation". In *Proceedings of the twentieth ACM symposium on Operating systems principles*, 1–2. ACM.

Hannon, C., J. Yan, and D. Jin. 2016. "DSSnet: A Smart Grid Modeling Platform Combining Electrical Power Distribution System Simulation and Software Defined Networking Emulation". In *Proceedings of the 2016 annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation*, 131–142. ACM.

IEEE 2012. "1815-2012 - IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3)".

Lamps, J., V. Adam, D. M. Nicol, and M. Caesar. 2015. "Conjoining Emulation and Network Simulators on Linux Multiprocessors". In *Proceedings of the 3rd ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM PADS '15, 113–124. New York, NY, USA: ACM.

Lamps, J., D. M. Nicol, and M. Caesar. 2014. "TimeKeeper: a lightweight Virtual Time System for Linux". In *Proceedings of the 2nd ACM SIGSIM/PADS conference on Principles of advanced discrete simulation*, 179–186. ACM.

Lee, R. M., M. J. Assante, and T. Conway. 2016. "Analysis of the Cyber Attack on the Ukrainian Power Grid". *SANS ICS Report*.

libpcap Developers 2016. "The Libpcap File Format". https://wiki.wireshark.org/Development/LibpcapFileFormat.

Mahoney, M. V., and P. K. Chan. 2003. "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection". In *International Workshop on Recent Advances in Intrusion Detection*, 220–237. Springer.

McHugh, J. 2000. "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 Darpa Intrusion Detection System Evaluations as Performed by Lincoln Laboratory". *ACM Transactions on Information and System Security (TISSEC)* 3 (4): 262–294.

Mininet 2016. "Mininet Virtual Network Tool". http://mininet.org/.

Morris, T. H., A. K. Srivastava, B. Reaves, K. Pavurapu, S. Abdelwahed, R. Vaughn, W. McGrew, and Y. Dandass. 2009. "Engineering Future Cyber-physical Energy Systems: Challenges, Research needs, and Roadmap". In *North American Power Symposium (NAPS), 2009*, 1–6. IEEE.

Pang, R., M. Allman, V. Paxson, and J. Lee. 2006. "The Devil and Packet Trace Anonymization". *ACM SIGCOMM Computer Communication Review* 36 (1): 29–38.

Power World 2016. "Power World Tool". http://www.powerworld.com/.

RTDS 2016. "Real Time Digital Simulation". http://www.rtds.com/.

Saran, A., S. K. Palla, A. K. Srivastava, and N. N. Schulz. 2008. "Real-time Power System Simulation using RTDS and NI PXI". In *Power Symposium, 2008. NAPS'08. 40th North American*, 1–6. IEEE.

Shiravi, A., H. Shiravi, M. Tavallaee, and A. A. Ghorbani. 2012. "Toward developing a Systematic Approach to Generate Benchmark Datasets for Intrusion Detection". *Computers & Security* 31 (3): 357–374.

Sommer, R., and V. Paxson. 2010. "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection". In *Security and Privacy (SP), 2010 IEEE Symposium on*, 305–316. IEEE.

Sommestad, T., and J. Hallberg. 2012. "Cyber Security Exercises and Competitions as a Platform for Cyber Security Experiments". In *Nordic Conference on Secure IT Systems*, 47–60. Springer.

Tavallaee, M., N. Stakhanova, and A. A. Ghorbani. 2010. "Toward credible Evaluation of Anomaly-based Intrusion-Detection Methods". *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40 (5): 516–524.

AUTHOR BIOGRAPHIES

VIGNESH BABU is a Ph.D. candidate in Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign (UIUC). His research interests include cyber security modelling for Smart Grid systems. Email: babu3@illinois.edu.

RAKESH KUMAR is currently a Ph.D. candidate at UIUC studying network resiliency and security. Email: kumar19@illinois.edu

HOANG HAI NGUYEN is a Ph.D. candidate in the Department of Electrical and Computer Engineering at UIUC. His research interests include cyber-physical systems and security. Email: hnguye11@illinois.edu

DAVID M. NICOL is the Franklin W. Woeltge Professor of Electrical and Computer Engineering at UIUC, and Director of the Information Trust Institute. He is the PI for two recently awarded national centers for infrastructure resilience: the DHS-funded Critical Infrastructure Reliance Institute, and the DoE funded Cyber Resilient Energy Delivery Consortium. His research interests include trust analysis of networks and software, analytic modeling, and parallelized discrete-event simulation, research which has lead to the founding of startup company Network Perception, and election as Fellow of the IEEE and Fellow of the ACM. He is the inaugural recipient of the ACM SIGSIM Outstanding Contributions award. He received the M.S. (1983) and Ph.D. (1985) degrees in computer science from the University of Virginia, and the B.A. degree in mathematics (1979) from Carleton College. Email: dmnicol@illinois.edu.

KARTIK PALANI is a Ph.D. candidate in Computer Engineering at UIUC. His research focuses on security for the smart power grid and the Internet of Things. Email: palani2@illinois.edu

ELIZABETH REED is a Ph.D. candidate in the ECE department at UIUC. Her research interests include cyber security and computer architecture. Email: ereed@illinois.edu