DefRec: Establishing Physical Function Virtualization to Disrupt Reconnaissance of Power Grids' Cyber-Physical Infrastructures

Hui Lin Jianing Zhuang
University of Nevada, Reno
{hlin2, jzhuang}@{unr, nevada.unr}.edu

Yih-Chun Hu
University of Illinois, Urbana-Champaign
yihchun@illinois.edu

Huayu Zhou
University of Nevada, Reno
hzhou@nevada.unr.edu

***Abstract*—Reconnaissance is critical for adversaries to prepare attacks causing physical damage in industrial control systems (ICS) like smart power grids. Disrupting reconnaissance is challenging. The state-of-the-art moving target defense (MTD) techniques based on mimicking and simulating system behaviors do not consider the physical infrastructure of power grids and can be easily identified.**

To overcome these challenges, we propose physical function virtualization (PFV) that "hooks" network interactions with real physical devices and uses these real devices to build lightweight virtual nodes that follow the actual implementation of network stacks, system invariants, and physical state variations in the real devices. On top of PFV, we propose DefRec, a defense mechanism that significantly increases the effort required for an adversary to infer the knowledge of power grids' cyber-physical infrastructures. By randomizing communications and crafting decoy data for virtual nodes, DefRec can mislead adversaries into designing damage-free attacks. We implement PFV and DefRec in the ONOS network operating system and evaluate them in a cyber-physical testbed, using real devices from different vendors and HP physical switches to simulate six power grids. The experimental results show that with negligible overhead, PFV can accurately follow the behavior of real devices. DefRec can delay adversaries' reconnaissance for more than 100 years by adding a number of virtual nodes less than or equal to 20% of the number of real devices.

I. Introduction

Reconnaissance is crucial to an adversary's preparation for an attack on industrial control systems like smart power grids (ICS) [77]. By obtaining in-depth knowledge of physical processes, adversaries can determine "attack-concept" operations to cause devastating physical disruptions without raising alarms [10], [17]. For the attack on a Ukrainian power plant that caused a blackout affecting 225,000 residents [37], [38], security analysis directly indicates that "the strongest capability of the attackers was ... to perform reconnaissance operations required to learn the environment." Reconnaissance allows adversaries to design attack strategies that cause physical damage (e.g., compromising measurement data or maliciously turning off switches).

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.24365
www.ndss-symposium.org

Compared to passive intrusion detection systems (IDS), which detect attacks after malicious activities, preemptive approaches that can disrupt reconnaissance before an adversary starts to inflict physical damage are highly desirable. *First*, by preventing reconnaissance on a critical set of physical data, we can cover a wide spectrum of attacks, including unknown ones. Although adversaries may exploit different vulnerabilities and perform diverse malicious activities to cause different types of physical damage, they rely on the same physical processes from the target system to plan effective attack strategies. *Second*, we can detect and mislead attacks before adversaries execute their strategies and inflict damage. Detecting attacks at this early stage enables us to remove potential threats and prevent damage.

Despite all these benefits, there exists a big research gap to design practical and efficient anti-reconnaissance approaches. Current research, such as moving target defense (MTD) techniques, relies on mimicking and simulation of system behaviors and thus, has critical drawbacks. *First*, mimicking system behaviors can be easily detected [25] because it often poorly follows a complete system specification and lacks interactions with environments. *Second*, simulations of network infrastructure, often used in honeypots or honeynets [6], [11], [72], are based on a static specification. Even if the implementation is perfectly consistent with a protocol specification, it can still be different from the actual implementation of a real system, e.g., one with proprietary implementation in a real utility environment. *Last*, those state-of-the-art honeypot projects for ICS do not model physical processes and system-level behaviors, e.g., latencies of executing control commands and variations of physical states, which can vary at different locations and times. Adversaries can exploit those state variations to identify the simulation [18].

A. Contributions

To overcome the drawbacks of mimicking and simulations, we propose the design of physical function virtualization (PFV) to build lightweight virtual nodes that follow the actual implementation of network stacks, physical state variations, and system invariants of real physical devices in power grids. PFV leverages a network control application based on software-defined networking (SDN) to "hook" network interactions with real devices and use them as network flows of virtual nodes (as shown in Figure 1).

PFV's Role. We position PFV as a complementary service

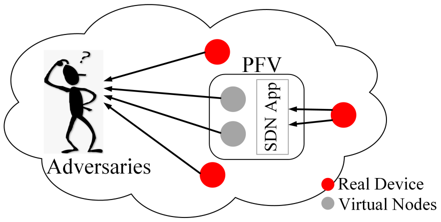


Fig. 1: **Design principle of PFV:** using SDN to interact with adversaries on behalf of virtual nodes based on interactions of real physical devices.

to existing security functionality, including IDSs and honeypots. Virtual nodes built by PFV cannot perfectly follow every aspect of physical devices from a large variety of vendors without formal coverage analysis (which we leave to future work). However, based on a concept completely different from network mimicking and simulation, PFV achieves lightweight virtualization on network interactions of physical devices, providing additional protection to complement passive IDSs:

- ***Considering practical threats with partial compromise.*** IDSs, especially anomaly-based ones, rely on a correct system model to identify anomalies; they require that no compromises occur while building system models. These assumptions are unrealistic and even dangerous in practice [65]. PFV considers a strong threat model in which adversaries have already *compromised power grids' networks* and *no trust* is needed on complicated SCADA (supervisory control and data acquisition) systems or substation configurations. PFV can potentially extend future IDSs to fight against practical and realistic threats.
- ***Increasing reconnaissance efforts.*** Virtual nodes follow the behavior of real devices at a fine-grained level that makes it computationally expensive, if not impossible, for adversaries to efficiently distinguish real devices from virtual nodes. Consequently, we expect PFV can significantly increase adversaries' reconnaissance efforts. Compared to high false detection rates in anomaly-based IDSs [65], PFV can help to reduce the rate of successful reconnaissance by at least three orders of magnitude (as shown in Section VII-A).
- ***Regaining computational advantages for defense mechanisms.*** Network packets from virtual nodes provide additional misleading information for adversaries, while defense mechanisms can leverage accurate system information. In Section V, we show that PFV can even make vulnerable state estimation used in conventional SCADA systems robust against stealthy cyberattacks.

Based on PFV, we present DefRec, a specific defense mechanism to *significantly increase the reconnaissance efforts required to infer the knowledge of power grids' cyber-physical infrastructures* without affecting legitimate applications that already know the actual power grid configurations (e.g., the identities of real physical devices). DefRec specifies and implements two security policies: (i) obfuscate communications by adding random interactions with virtual nodes, introducing significant overhead for adversaries to identify real devices; and (ii) mix decoy data (from virtual nodes) with real data (from physical devices), based on which adversaries would design ineffective and easy-to-detect attacks (e.g., activities that access virtual nodes).

Even though we present the design of DefRec (and PFV) for power grids, the design concept can be extended to other ICSs. By adding parsers and encoders of protocols used in different ICS networks and profiling characteristics reflecting system invariants in those environments, PFV can monitor and manipulate network interactions with real devices without any proprietary instrumentation. Based on those adjustments in PFV, the first security policy in DefRec that randomizes communications with virtual nodes is also extensible. The second security policy relies on the control theoretical model of power grids. By using the model of physical processes in other ICSs, we can also implement the security policy for different utility environments.

The main contributions of the paper are:

- To the best of our knowledge, this is the first work to propose the concept of PFV (physical function virtualization) that builds virtual nodes following the actual implementation of network stacks, system invariants, and physical state variations of real physical devices.
- To the best of our knowledge, DefRec is also the first work that aims at increasing the difficulty for an adversary's reconnaissance to infer the knowledge of power grids' cyber-physical infrastructures, by randomizing communications and crafting decoy data for virtual nodes.
- For evaluation, we used real devices to implement both cyber and physical infrastructures of power grids: (i) real intelligent electronic devices (IED) from three different vendors, (ii) five HP SDN-compatible switches to build six communication networks containing up to 124 nodes, and (iii) simulations of six real-world power grids containing up to 1,000 substations.

Our experimental results show that DefRec, together with PFV, is highly effective at disrupting reconnaissance while introducing negligible performance overhead:

- PFV can build virtual nodes closely following the implementation of real devices, including complete network stacks, system invariants such as latencies of executing commands, and runtime variations of physical states.
- DefRec can significantly increase the reconnaissance efforts to identify real devices, e.g., delaying adversaries for at least 100 years by adding a number of virtual nodes less than or equal to 20% of the number of real devices.
- DefRec can craft decoy data to further increase adversaries' reconnaissance on power grids' physical infrastructure, with less than 0.5% false negatives.
- Performance overhead is negligible: (i) PFV can efficiently construct virtual nodes with the goodput of at least 1.5 Mbps (sufficient for a power grid with 30,000 measurements), and (ii) DefRec introduces less than 3% overhead on existing network communications and power grid operations.

II. Design Objectives

In this section, we first present background information on power grids. Then we describe our design principles based on the threat model considered in this work.

A. Power Grid Basics

A power grid is an ICS, in which generators supply power to load demands over a wide geographical area. The generators and load demands are connected by transmission lines in a complex topology, often referred to as a *transmission network*. In the graphical representation of a transmission network (e.g., in Figure 3), we use a *bus* to represent a substation, where generators or load demands are deployed. In each bus and transmission line, we can have physical measurement data, including voltage, current, power consumption, and generation.

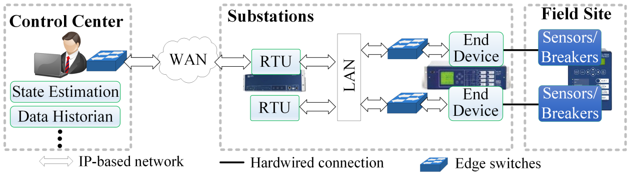


Fig. 2: **Hierarchical network infrastructure of power grids.**

In Figure 2, we show a hierarchical communication network used by power grids. A control center uses an IP-based *control network* to retrieve data from substation devices periodically; this process is also known as *data acquisition*. Based on the retrieved data, the control center uses *state estimation* to determine the physical state of power grids (details in Section V-A1).

For further discussion, we use the following definitions:

Definition 1. End Devices: Intelligent electronic devices (IED) located at the end of a communication path connecting the control center and substations.

End devices connect to sensors or circuit breakers through hardwired connections in their downstream communications. In their upstream communications, multiple end devices connect to a higher-level IED, e.g., RTUs (remote terminal units), which forwards information (e.g., aggregated measurements or commands) to/from the control center.

Definition 2. Edge Switches: Network switches located at the first or the last hop of a communication path that connects the control center and end devices.

B. Design Objective & Threat Model

Assumptions on Adversaries' Capability. We assume that adversaries penetrating a control network have limited knowledge of network configurations and physical data. We assume that adversaries can compromise any computing devices connected to the control network; however, they are not able to obtain knowledge of a whole power grid based on data collected by those compromised devices. Unlike previous IDS designs [5], [9], [41], we do not require the trust of SCADA systems in a control center and end devices in substations (except a few devices used by PFV, see following paragraphs).

Compared to external adversaries learning power grids through coarse-grained publicly available data, these "insider" adversaries are a more significant threat. For example, even though external adversaries can know the basic connection of substations through satellite pictures, they are not able to obtain connections of physical devices and measurements these devices collect, which reflect a much more complicated graph required to design effective attacks [75]. The insider adversaries can perform reconnaissance in such fine granularity, which can only be obtained through internal control networks (as seen in real attack incidents [37], [38]).

For clarity, we classify adversaries' attack capabilities into three types. ***Passive attacks*** monitor network traffic to obtain the knowledge of power grids' cyber-physical infrastructures. ***Proactive attacks*** achieve the same goal by using probing messages to trigger responses from real devices or virtual nodes. ***Active attacks*** directly manipulate network traffic, including dropping, delaying, compromising existing network packets, or injecting new packets. *Passive and proactive attacks are common techniques used by adversaries to perform reconnaissance, while active attacks are used to issue attack-concept operations to cause physical damage.*

DefRec's Objective. DefRec's objective is to disrupt and mislead adversaries' reconnaissance based on ***passive*** and ***proactive*** attacks, such that their ***active*** attacks become ineffective. Reconnaissance is a necessary step for "targeted attacks" in ICSs [8], [39], which are more frequently appearing in real utilities [17], [37], [38] and are becoming a critical and damaging threat for ICSs, including power grids. In previous *targeted attacks*, adversaries have used in-depth knowledge of the target systems (obtained through reconnaissance) to stealthily deliver malicious attacks. We specify our anti-reconnaissance objectives (RO) as follows:

- ***RO1: for passive attacks on a control network***, we aim at significantly lengthening the time required by adversaries to successfully learn the knowledge of the control network.
- ***RO2: for proactive attacks on a control network***, we aim at revealing adversaries' existence with a high probability and isolating the compromised devices from the network.
- ***RO3: for physical knowledge obtained by passive or proactive attacks***, we aim at leveraging intelligently crafted decoy data to mislead adversaries into designing ineffective attacks.

To achieve these objectives, we add only the components of PFV in the trusted computing base (TCB, highlighted in Figure 3), which includes lightweight SDN controller applications, network switches to which the SDN controller is attached, a few real physical devices (referred to as "seed" devices) from which we build virtual nodes, and the communication channels between the aforementioned components. Those communication channels, such as the one connecting SDN applications and the seed devices, can be protected by SSL/TLS, to prevent adversaries' eavesdropping. These assumptions are typical in work that studies attacks targeting SDN [70], [74]. Compromising SDN controllers or edge switches requires adversaries to invoke or inject certain network events [70]. Because DefRec performs simple activities, e.g., sending out network traffic on behalf of virtual nodes, we can ensure the assumptions made in the threat model by monitoring anomaly events from/to SDN controllers [15].

DefRec's objective is to increase adversaries' reconnaissance efforts. Even if adversaries learn the identities of virtual nodes and real devices, we can reconfigure power systems, e.g., changing IP addresses of real devices, to disrupt their future reconnaissance. In addition, adversaries still need to face existing defense mechanisms, such as IDSs designed for *active* attacks. We believe that PFV can also help the design of IDSs for active attacks, which we leave to future work.

For Attacks Requiring Little or No Reconnaissance. DefRec does not focus on attacks that require no or little

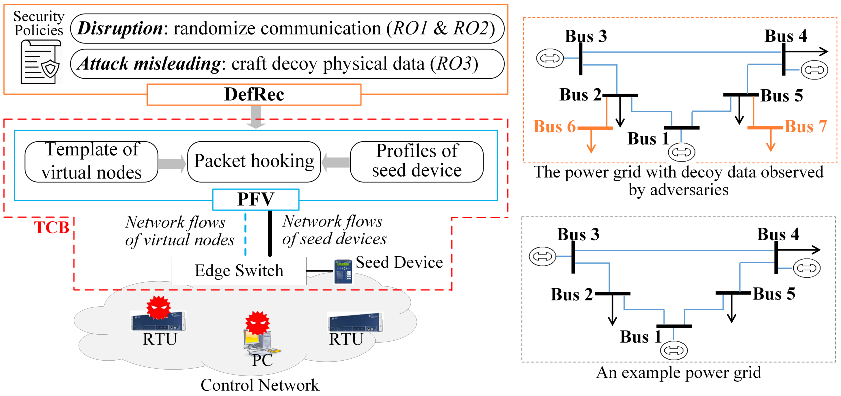

Security Policies
Disruption: randomize communication (RO1 & RO2)
Attack misleading: craft decoy physical data (RO3)
DefRec
Template of virtual nodes
Packet hooking
Profiles of seed device
PFV
TCB
Network flows of virtual nodes
Network flows of seed devices
Edge Switch
Seed Device
RTU
PC
RTU
Control Network
Bus 3
Bus 4
Bus 2
Bus 5
Bus 6
Bus 1
Bus 7
The power grid with decoy data observed by adversaries
Bus 3
Bus 4
Bus 2
Bus 5
Bus 1
An example power grid

Fig. 3: **Design overview:** (i) PFV constructs virtual nodes that follow the actual implementations of seed devices; and (ii) DefRec specifies security policies based on PFV to randomize communications and to craft decoy data for virtual nodes.

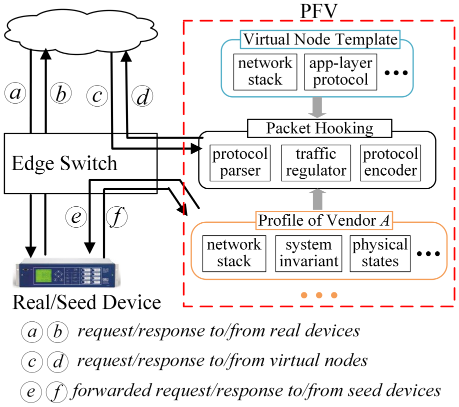


Fig. 4: **Components of PFV:** hook network interactions with real devices based on virtual nodes template and runtime profiling.

reconnaissance, including adversaries that (i) target applications storing or using all network configurations and physical data in a single location, (ii) have already compromised a large number of physical devices, (iii) perform attacks through random *physical* disruptions or data compromises, and (iv) have a direct access to a large number of physical devices, e.g., through Internet-of-things (IoT) [64]. For the first case, it is necessary to dedicate specialized protections on those critical applications, such as running them in an isolated and trusted environment like Intel SGX [12], which has already been applied in critical power grid applications [32], [62].

For the second type of attacks, adversaries can introduce physical damage without reconnaissance. As shown in [41], [42], if adversaries can compromise 50% of physical devices, they can very likely introduce physical damage through random *active* attacks (probability is more than 60%). As normal control operations seldom involve such a large number of physical devices, this type of attack can trigger alerts under existing protection mechanisms.

For the third and fourth types of attacks that require little reconnaissance, we can still remedy them by enhancing PFV with different security policies. The insight is that in random or IoT-based attacks, adversaries rely on numerous control commands to change power grids' physical state, mainly through *active* attacks. Without knowing the identities of real physical devices, their malicious activities can access virtual nodes and raise alerts. We leave the formal analysis of detecting those attacks based on PFV to future work.

For Attacks on Data Privacy. Disrupting adversaries' reconnaissance is different from protecting the privacy of physical data. The proposed security policies are to achieve RO1–RO3, which may not be consistent with policies to prevent the leaking of physical data. We leave the design of privacy-preserving policy to future work.

III. Design Overview of DefRec Based on PFV

In Figure 3, we present the design of DefRec, including the components of PFV and two security policies implemented on top of it (details are presented in Section IV and Section V). We position PFV as a complementary service to defense mechanisms, such as the design and implementation of the proposed security policies to disrupt and mislead adversaries' reconnaissance. We believe that PFV's functionality is not limited to DefRec, but can be used in other security solutions.

A. Components of PFV

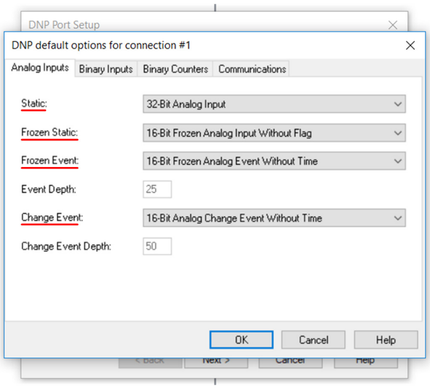
The objective of PFV is to build lightweight virtual nodes that follow the implementation of network stack, system invariants, and physical state variations of real devices. In Figure 4, we present three components of PFV in detail.

Virtual Node Templates. We use these templates to contain basic configurations of the target control networks. For example, the templates include network stack information, such as IP addresses that can be assigned to virtual nodes, and the specification of application-layer protocols used to deliver physical data and control operations. Configurations stored in the templates are not specific to the context of a power grid.

Profile of Seed Devices. We select a few *end devices* as seed devices and profile three aspects of each.

- *The actual implementation of network stack* can be different from the protocol specification. For example, the DNP3 protocol used in power grids specifies 37 function codes [29], but the SEL 751A relay used in our experiment implements only 14 of them [60].
- *System invariants* refer to the characteristics that can be used to identify or fingerprint real devices, such as the latency of executing control commands [18].
- *Physical state variations* usually fall in a deterministic range for a specific power system, such as voltage magnitudes varying within the range of ±5% around a nominal value [20].

The device profile includes a range of variations for a certain property observed at runtime (e.g., command execution time) and the probability distribution over that range. We only need to select one seed device to represent each vendor or model, based on which we profile the runtime behavior. The device profile makes network flows from virtual nodes follow the probabilistic behavior of real devices, rather than replicating the same pattern. In this paper, DefRec focuses on the reconnaissance of power grids' applications. As such, we build virtual nodes following these three aspects of seed devices. When using DefRec for other applications, we can profile other application-specific knowledge.



Static: 32-Bit Analog Input
Frozen Static: 16-Bit Frozen Analog Input Without Flag
Frozen Event: 16-Bit Frozen Analog Event Without Time
Event Depth: 25
Change Event: 16-Bit Analog Change Event Without Time
Change Event Depth: 50

Fig. 5: **DNP3 configuration in Schneider Electric ION 7550 power meters:** specifying four data formats for analog inputs.

We can practically implement the profiles of seed devices with a little storage overhead. We do not rely on proprietary instrumentation on physical devices. Instead, we can select existing IEDs as seed devices by reusing spare Ethernet ports (e.g., the SEL 751A relay can include up to three Ethernet ports [60]). Then we can extract knowledge from network packets (ⓔ and ⓕ in Figure 4) through the "protocol parser" in the *packet hooking* component. Because a substation tends to deploy devices from the same vendors to reduce configuration complexity [18], we expect to profile a small number of models in a substation. Also, many physical devices include a small number of configurations for a few deterministic functions. For example, as shown in Figure 5, the Schneider Electric ION7550 power meter [58] used in our experiments only allows at most four data formats for analog inputs, binary inputs, and binary counters (due to space limitations, only the tab of "analog inputs" is shown).

Packet Hooking. This component uses SDN's programmability to hook network packets from seed devices; it tailors these packets based on the information from device profiles and virtual node templates, and uses the resultant packets as the network flows from virtual nodes.

As shown in Figure 4, when network packets reach an edge switch, a *protocol parser* extracts header information from the packets; a *traffic regulator* redirects them to a seed device if their destinations are virtual nodes (network flow ⓒ and ⓔ in Figure 4). Upon receiving the forwarded packets, the seed device responds (ⓕ). The responding packets serve two purposes: (i) building the device profile and (ii) serving as the input to a *protocol encoder*, which tailors the network packets according to the system invariants and physical state variations profiled before. The tailored packets, which are not deterministic but follow the same probabilistic properties of seed devices, are sent out by virtual nodes as the responses for the original request (ⓓ). The responses from seed devices reflect the actual implementation of the whole network stack, including the TCP/IP implementation. Consequently, virtual nodes are able to respond to lower layer network probing, e.g., ARP requests.

There are two requirements for the packet hooking component so that virtual nodes follow the runtime behavior of seed devices without causing physical damage and revealing the real physical state of a power grid.

- *Tailor application-layer payloads.* A *protocol encoder* tailors the application-layer payloads of network packets sent out by virtual nodes to (i) avoid leaking real physical data and (ii) introduce entropy (according to the device profile) to the data sent out by virtual nodes. A method for tailoring the payloads is beyond the range of PFV design. In Section V, we craft decoy data for virtual nodes to mislead adversaries.
- *Redirect control operation without physical impact.* If a control operation (e.g., turn off a circuit switch) reaches a virtual node, we redirect it to a seed device, connecting to a switch that is already turned off, such that the operation introduces no physical impacts on the power grid.

To reduce the implementation complexity and runtime overhead (caused by SDN), we further enhance the packet hooking component with two design options.

- *Take advantage of edge switches.* As shown in Figure 4, we choose to attach PFV components to edge switches, which brings two benefits. *First*, we can reduce controller-switch latency. The latency of fewer than 2 milliseconds (found in our experiments and specified in IEC 61850 [23]) is within the range of the system invariant of seed devices. *Second*, we can reduce the number of switches in the communication path connecting the seed device and the packet hooking component. Consequently, we only need to include the edge switch in the TCB.
- *"Cache" interactions with seed devices.* Because communications in power grids experience little variation and are repetitive in nature, we can optionally (determined by system administrators) cache requests and the corresponding responses. For example, we only need to cache "request-response" pairs for around fourteen configurations for the ION 7550 power meter shown in Figure 5. When an incoming request matches an entry in the cache, we directly respond according to the profile of the device model without redirecting it to seed devices.

  "Caching" can significantly reduce the frequency of the interaction with seed devices, while making responses from the virtual nodes still follow the network implementation, system invariants, and physical state variations of real devices. If we can perform sufficient caching in advance, we can completely avoid the interaction with seed devices, allowing us to further remove seed devices from the TCB.

Implementation. We can implement PFV based on any network manipulation techniques. However, the network programmability and visibility enabled by SDN can significantly benefit the design and implementation of PFV and security policies. Also, SDN-enabled networks are being designed and deployed for power system substations [59], which make it feasible to deploy PFV in real utility environments (in our testbed, we have successfully implemented them with real physical devices).

Built on lightweight SDN controller applications, PFV requires no modifications to (i) existing control operations from energy management systems, (ii) physical configurations of substations, or (iii) existing network routing/forwarding configurations in control networks.

B. Security Policies based on PFV

Based on PFV, DefRec specifies security policies to achieve anti-reconnaissance objectives defined in Section II-B. As shown in Figure 3, to achieve RO1 and RO2 related to the reconnaissance of power grid networks, we present in Section IV a *disruption* policy that randomizes network communications. To achieve RO3, in Section V, we propose an *attack-misleading* policy that crafts decoy data for virtual nodes. Consequently, adversaries collect misleading information about a power grid different from the one under protection (e.g., the power grid with virtual nodes highlighted in orange in Figure 3). Based on such information, adversaries design ineffective attacks.

IV. Disruption Policy: Randomize Communications

The disruption policy deployed in DefRec is to achieve the anti-reconnaissance objectives RO1 and RO2. To disrupt *passive* attacks in RO1, we randomize network packets issued to both real devices and virtual nodes, significantly increasing the time for adversaries to stealthily identify real devices (Section IV-A). To disrupt *proactive* attacks in RO2, we introduce randomness when responding to probing of virtual nodes, revealing adversaries' existence with a high probability while reducing the information an adversary can learn (Section IV-B).

A. Randomize Request Patterns

Adversaries can use *passive* attacks to stealthily identify real physical devices, if we issue requests only to real devices (such as @ in Figure 4). To address this problem, we randomize request patterns as follows:

- *Add requests to randomly selected virtual nodes.* Because accessing this set of virtual nodes is part of the disruption policy, we regard them as "accessible virtual nodes" and accessing them will not raise alerts. When we change the addresses of accessible virtual nodes, we always leave the addresses of a subset of virtual nodes unchanged. Otherwise, adversaries can conveniently identify real devices because their addresses remain unchanged.
- *Issue requests to randomly selected real devices.* This randomization activity further balances the number of requests sent to real devices and requests to accessible virtual nodes. It prevents adversaries from identifying real devices based on the biased distribution of the number of network packets sent to the real devices.

  Sending requests to randomly selected real devices has become feasible as modern power grids deploy a large number of devices to collect redundant measurements. Consequently, we can issue requests to randomly selected real devices as long as they collect the set of "necessary measurements" (or basic measurements) [45]. As the set of necessary measurements are not fixed but vary with system state, the requests issued to the real devices are continuously randomized. Currently, many work exploits this property to reduce communication overhead with negligible impact [21], [40], [47]. In Appendix A, we also show that such impact is negligible, as the accuracy of common control operation is at least 99.8%.

Security Argument. We assume that there are n real devices and m virtual nodes, which include m_1 accessible nodes. The detailed derivation and evaluation of the following security argument can be found in Appendix A. The probability that an adversary will successfully identify all accessible virtual nodes is $(\binom{n}{n'} \times \binom{n'+m_1}{m_1})^{-1}$, if the adversary issues requests to n' randomly selected real devices and m_1 randomly selected virtual nodes; this probability decreases exponentially with n' and m_1.

B. Probabilistic Dropping Protocol to Isolate Proactive Attacks

Adversaries can perform *proactive attacks* to probe physical devices or virtual nodes and use the responses to identify real devices. Probing *accessible* virtual nodes always results in responses. In this section, we present a procedure to handle proactive probing destined to other "inaccessible" virtual nodes, which is suspicious, as the requests are neither from legitimate applications nor DefRec. However, directly isolating the probing machine can immediately reveal the identity of an inaccessible virtual node.

We propose a "probabilistic dropping" protocol that significantly reduces the effectiveness of adversaries' probing activities, making it info-theoretically challenging for an adversary to determine which devices are real.

- We label an identity as *suspicious* if it accesses an inaccessible virtual node and isolates this suspicious identity from the network (e.g., dropping following incoming packets) with the probability p_0.
- For the k-th access from the suspicious identity to either virtual or real devices ($k \geq 1$), we isolate the suspicious identity with a probability p_k. If the suspicious identity is not isolated on this access, it will receive a response.
- We set a threshold δ for the maximum number of accesses allowed for the suspicious identity.

We model the event to isolate a suspicious identity after the access to an inaccessible virtual node as a series of biased coin flips with increasing probability. The probability that the suspicious identity is isolated at its k-th access is $Q_k = p_k \times \prod_{j=0}^{k-1}(1-p_j)$ for $1 \leq k \leq \delta$.

Security Argument. Following the assumption that there are n real devices and m_2 inaccessible virtual nodes, we present a brief security argument here and leave the detailed derivation and evaluation to Appendix A.

When an adversary finds that her compromised identity is isolated from the network after a probing, she can guess on which previous access (among up to $\delta + 1$ accesses) she probed an inaccessible virtual node. We want to make each previous access have an equal chance of leading to the isolation of the suspicious identity. In other words, we want to make $p_0 = Q_0 = Q_1 = \cdots = Q_\delta$. Then $p_k = p_0/(1 - k \cdot p_0)$ for $0 \leq k \leq \delta$. To make $p_k < 1$, we need to determine p_0 such that $(1 - k \cdot p_0) > p_0$ always holds, which is equivalent to $p_0 < 1/(1+\delta)$. Under this condition, the probability that an adversary has accessed δ real devices is $S_\delta = p_0^\delta \prod_{k=1}^{\delta} \binom{n+m_2-1}{n-k+1}^{-1}$. If the adversary accesses all remaining real devices under this protocol, the probability that they can obtain all real measurements through proactive probing will be at least $S_{all} = S_\delta^{\lceil \frac{n}{\delta} \rceil}$.

Security-Performance Trade-off Argument. With more virtual nodes, it is more difficult for adversaries to identify real devices. Meanwhile, with more virtual nodes, we redirect more packets to seed devices, which handle more requests compared to the case without DefRec.

To meet a security-performance trade-off, we can adjust the design parameters (i.e., m_1, m_2, and δ). For example, we can increase the number of virtual nodes by varying their identities, e.g., IP addresses, even infrequently, such as once every multiple days. As we randomize the request patterns for both real devices and virtual nodes, an adversary's chances of successfully identifying real devices can drop significantly, e.g., less than 10^{-5} even the number of virtual nodes equal to 5% of the number of real devices (shown in Section VII-A). If we use caching in PFV, we can further reduce the amount of requests that are actually forwarded to seed devices.

V. Attack-Misleading Policy: Craft Decoy Data

In addition to disrupting the reconnaissance on network configurations, we further use the "attack-misleading" policy to achieve RO3: disrupting adversaries' reconnaissance on power grids' physical infrastructure by providing misleading knowledge. The core of this policy is an algorithm that crafts decoy data for virtual nodes. Without careful design, data piggybacked by network packets from virtual nodes can still allow adversaries to learn grids' physical knowledge, e.g., physical topology and measurements.

The decoy data construction algorithm takes the following inputs: the states of real physical devices, the number of virtual nodes, and the topology graph representing their connectivity. We put those inputs into a power system's mathematical representation to obtain decoy states or decoy data for virtual nodes, which follow the normal variations observed in real physical devices. For example, in Figure 3, we add four virtual nodes to represent substations with load units (Bus 6 and Bus 7) and two transmission lines connecting them to the power system; we use the algorithm to obtain decoy data, such as the impedance of the added lines and power consumption in the added buses. We can implement the decoy data construction algorithm in any power grid analysis tools running on general-purpose computers, such as MATPOWER that we use for our evaluations [79].

There are two requirements for constructing decoy data: (i) mislead adversaries into designing ineffective strategies (Section V-A3); and (ii) follow the physical model of power grids to avoid suspicion from adversaries (Section V-A4). In this section, we focus on crafting decoy data for false data injection attacks (FDIAs). FDIAs, originally presented in [42], can significantly downgrade the accuracy of state estimation and the performance of many power grid applications. The well-established theoretical model in this attack can help us to formally prove the effectiveness of decoy data. In Appendix B, we discuss how to apply the decoy data construction algorithm for FDIAs in attacks targeting other control operations, showing the potential impacts of the algorithm.

A. Decoy Data Construction to Mislead FDIAs

1) Background in State Estimation: Power grids use a complicated mathematical model to correlate data collected from substations. To be concrete, we use the DC power flow analysis model shown in Equation (1) to relate state variables to sensor measurements.

$$\mathbf{z} = H\mathbf{x} + \mathbf{e} \qquad (1)$$

where $\mathbf{z} = [z_1, z_2, \ldots, z_q]^T$ is a column vector representing q measurements ($[\cdot]^T$ represents the transpose of a vector); $\mathbf{x} = [x_1, x_2, \ldots, x_p]^T$ represents p physical state or p phasor angles at all buses; $\mathbf{e} = [e_1, e_2, \ldots, e_q]$ is the collection of q measurement errors; and H is a q-by-p measurement matrix. There are many sensor measurements, e.g., voltage, currents, and power, that can be used to represent the physical state of a power grid. For historical reasons, "sensor measurements" refer to data that can be efficiently measured by legacy sensors, e.g., active power injected at each substation or active power delivered by transmission lines (measuring current can be very expensive). "State variables," indirectly estimated via "state estimation," refer to voltage angles at each substation, as their variations are sensitive to the grid stability [20], [45].

We follow the derivation in [35] to determine the measurement matrix H, whose entries are determined by the topology of the transmission network, the susceptance of each transmission line (i.e., the imaginary part of admittance, which is the inverse of impedance), and the placement of sensors. We assume that a line sensor measures the active power flow of each transmission line and a bus sensor measures the net active power injected into that bus. Therefore, the matrix H is generated as follows. The row of H associated with the sensor of transmission line k connecting bus i and j is

$$H_{k:(i,j)} = [\mathbf{0} \qquad \underset{(i\text{-th entry})}{B_{ij}} \qquad \mathbf{0} \qquad \underset{(j\text{-th entry})}{-B_{ij}} \qquad \mathbf{0}] \qquad (2)$$

where the index $k : (i, j)$ represents a transmission line connecting buses i and j, $\mathbf{0}$ is a zero-vector of an appropriate size, and B_{ij} is the susceptance of the transmission line. In this row of H, B_{ij} and $-B_{ij}$ are the i-th and j-th entries.

The row of H associated with the sensor at bus j is

$$\sum_{i\text{: incident to } j} H_{k:(i,j)} \qquad (3)$$

where i refers to the index of a bus incident to bus j.

When measurement error $\mathbf{e}$ follows a normal distribution with zero mean, the estimation of state variable $\hat{\mathbf{x}}$ can be obtained through statistical criteria, e.g., the weighted least-square criterion. When estimating $\hat{\mathbf{x}}$, state estimation detects and removes bad sensor measurements to ensure that the estimated state variable comes "closer" to the actual state. Specifically, if an L_2-norm of the data residual used by state estimation is larger than a threshold, i.e.,$||\mathbf{z} - H\hat{\mathbf{x}}|| > \tau$, state estimation declares the presence of bad data and raises an alert.

2) Attack Preparation: In FDIAs, adversaries' objective is to compromise measurements $\mathbf{z}$ so that the estimated state variable $\hat{\mathbf{x}}$ becomes a different value $\hat{\mathbf{x}}_a = \hat{\mathbf{x}} + \mathbf{c}$, which is "further" away from the actual state by $\mathbf{c}$, without triggering alerts in state estimation [35], [42].

To make FDIAs successful, adversaries can inject an attack vector $\mathbf{a}$ into the original measurement $\mathbf{z}$. With the *full knowledge of H*, adversaries can construct $\mathbf{a}$ such that $\mathbf{a} = H\mathbf{c}$ and determine the corresponding compromised measurements

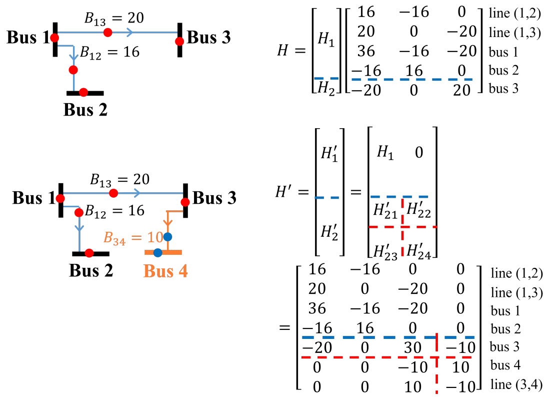


Fig. 6: **Misleading FDIAs in a 3-bus power system.** The decoy data is highlighted in orange.

as $\mathbf{z}_a = \mathbf{z} + \mathbf{a}$. In that case, the L_2-norm of the measurement residual becomes:

$$||\mathbf{z}_a - H\hat{\mathbf{x}}_a|| = ||\mathbf{z} + \mathbf{a} - H(\hat{\mathbf{x}} + \mathbf{c})|| = ||\mathbf{z} - H\hat{\mathbf{x}}|| \leq \tau \quad (4)$$

Because L_2-norm of measurement residual of compromised measurement $\mathbf{z}_a$ is less than τ, state estimation raises no alerts on the compromised measurements.

3) Crafting Decoy Data to Mislead FDIAs: Because adversaries rely on measurement matrix H to determine compromised measurements, we add virtual nodes to build a new power grid with a different measurement matrix; the compromised measurements determined based on this measurement matrix (which is decoy data) would always raise alerts in state estimation (see the following derivations).

Grid Configurations based on Virtual Nodes. When adding virtual nodes into a power system, dimensions of state variable $\mathbf{x}$ and sensor data $\mathbf{z}$ increase correspondingly. We represent new state variable and sensor data as the combination of real data and decoy data (from virtual nodes): $\mathbf{x}' = [\mathbf{x}^T \;\; \mathbf{x}_d{}^T]^T$ and $\mathbf{z}' = [\mathbf{z}^T \;\; \mathbf{z}_d{}^T]^T$, where $(\cdot)_d$ represents decoy data.

After decoy data is added, the measurement matrix also changes from H to H'. We use Figure 6 to show the construction of H'. In this figure, we mark the value of susceptance of each transmission line in a 3-bus system. We add two virtual nodes: Bus 4 and a transmission line connecting it to Bus 3. The decoy data of interest is B_{34}, the susceptance of line (3, 4) (other decoy data, e.g., active line power and bus power are not needed in this discussion). In the figure, we associate each row of H and H' with the corresponding line or bus (at the end of that row), based on which entries in that row are determined according to formulas (2) and (3).

To better understand the relationship between H and H', we divide H into two sub-matrices H_1 and H_2. Sub-matrix H_2 corresponds to real physical components that are affected by decoy data, while H_1 corresponds to real components that are not. In the example shown in Figure 6, because we connect virtual Bus 4 to real Bus 3, H_2 includes a single row associated with Bus 3.

After we add Bus 4 and line (3, 4), matrix H changes to H' as follows. Each row of H'_1 is constructed by appending an appropriate number of 0 at the end of each row in H_1. This is because physical components associated with rows in H_1 are not affected by decoy data $\mathbf{x}_d$. In H_2, we first append two rows (i.e., rows 6 and 7 of H') corresponding to virtual Bus 4 and virtual transmission line (3, 4). Because of this virtual line, the rows associated with Bus 3 and Bus 4 in matrix H'_2 is changed correspondingly, per formula (3). To see those changes explicitly, we further divide H'_2 into four sub-matrices, i.e., H'_{21}, H'_{22}, H'_{23}, and H'_{24} in Figure 6, such that H'_{21} has the same dimensions as H_2.

How Adversaries Prepare FDIAs based on Decoy Data. With an attack objective being $\mathbf{c}'$, an adversary determines an attack strategy $\mathbf{a}'$ according to the condition $\mathbf{a}' = H'\mathbf{c}'$ based on decoy matrix H', where $\mathbf{a}' = [\mathbf{a}^T \;\; \mathbf{a}_d^T]^T$ and $\mathbf{c}' = [\mathbf{c}^T \;\; \mathbf{c}_d^T]^T$. Here, $\mathbf{a}_d$ and $\mathbf{c}_d$ are changes made on decoy data $\mathbf{z}_d$ and $\mathbf{x}_d$ correspondingly.

How FDIAs Become Ineffective. When state estimation of a power grid receives compromised measurements, it verifies the integrity of measurements using condition $\mathbf{a} = H\mathbf{c}$, which is different from the one (i.e., $\mathbf{a}' = H'\mathbf{c}'$) used by adversaries to determine the compromised measurements. While the adversary thinks she has designed a successful attack, the involvement of decoy data makes it detectable even based on existing state estimation.

Is it Possible for FDIAs to Bypass DefRec (False Negative)? We used mathematical representations of attack preparations of FDIAs to demonstrate that it is challenging, if not impossible, for FDIAs to bypass DefRec.

In Equation (5), we expand the condition that adversaries use to determine compromised measurements.

$$\mathbf{a}' = H'\mathbf{c}' \Rightarrow \begin{bmatrix} \mathbf{a} \\ \mathbf{a}_d \end{bmatrix} = \begin{bmatrix} H_1 & \mathbf{0} \\ H'_{21} & H'_{22} \\ H'_{23} & H'_{24} \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \mathbf{c}_d \end{bmatrix} \Rightarrow$$
$$\mathbf{a} = \begin{bmatrix} H_1 \\ H'_{21} \end{bmatrix} \mathbf{c} + \begin{bmatrix} \mathbf{0} \\ H'_{22} \end{bmatrix} \mathbf{c}_d$$
$$\mathbf{a}_d = [H'_{23}]\,\mathbf{c} + [H'_{24}]\,\mathbf{c}_d \quad (5)$$

To avoid alerts from state estimation used in power grids, FDIAs determined based on decoy data need to satisfy the condition $\mathbf{a} = H\mathbf{c}$. By putting Equation (5) into this condition, we have:

$$\begin{bmatrix} H_1 \\ H'_{21} \end{bmatrix} \mathbf{c} + \begin{bmatrix} \mathbf{0} \\ H'_{22} \end{bmatrix} \mathbf{c}_d = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} \mathbf{c} \quad (6)$$

Equation (6) indicates the necessary condition for adversaries to bypass DefRec, i.e., performing successful FDIAs even with injections of decoy data. In practice, this corresponds to one of the following two conditions, which are difficult to satisfy:

- ***Adversaries are forced to change their attack strategies,*** to satisfy the condition $H'_{21}\mathbf{c} + H'_{22}\mathbf{c}_d = H_2\mathbf{c}$. Even adversaries are successful to satisfy the condition, forcing them to change attack strategies can directly affect effectiveness and stealthiness of the original attacks. As the scale of power grids increases, dimensions of Equation 6 also increase, making the condition difficult to satisfy according to our evaluation (see Section VII-A3).

- ***A power grid fails to deploy sufficient sensors***. If there are no sensors deployed at the physical component associated with H_2 (e.g., Bus 3 whose sensor measurements are affected by decoy data), we have $H_2 = H'_{21} = H'_{22} = \Phi$, which is an empty matrix. This is very unlikely to happen, as modern power grids usually deploy many redundant sensors to ensure accurate state estimation [45].

4) Refine Decoy Data to Follow Physical Model: The initial values of decoy data determined in Section V-A3 may be invalid, i.e., they may fail to follow the mathematical model of a power grid, raising suspicions to adversaries. To solve this problem, we refine decoy data such that the combination of decoy and real data becomes valid, which makes them appear to be measured from a real power grid. In other words, the combination of decoy and real data raises no alerts in state estimation.

We use Algorithm 1 to refine decoy data by going through iterations of two operations: (i) putting decoy and real data into state estimation (lines 5 and 8), and (ii) using resultant errors to adjust the value of decoy data while leaving the value of real data unchanged (line 7). Typically, state estimation removes data with big measurement errors due to accidents like misconfigurations in sensors [45]. In Algorithm 1, we use errors differently: modify decoy data instead of removing them. Using the resultant errors to slightly adjust the values of decoy data makes them move closer to valid ones, still achieving the misleading objective (e.g., making Equation (6) challenging to hold).

B. Discussion: Decoy Data for Other Attacks

In this section, we used a theoretical model of FDIAs to demonstrate decoy data construction. In Figure 24 in Appendix B, we further show that by replacing the theoretical model with ones in other control operations, we can generalize decoy data construction for other attacks.

Towards Future Smart Grids. DefRec aims at disrupting adversaries who need global knowledge of a power grid. In future microgrid infrastructure, distributed energy resources, such as solar power, decentralize control operations in multiple regions [36]. Consequently, DefRec may become less effective against adversaries that restrict their malicious activities in a region. To solve this problem, we can deploy multiple DefRec instances independently in concerned regions to disrupt and mislead adversaries' preparations relying on knowledge of those regions.

Algorithm 1 Pseudocode of REFINEDECOY that refines decoy data such that the combination of decoy and real data is valid

```
1: Input: $\mathbf{z}^{init} = [\mathbf{z} \;\; \mathbf{z}_d^{init}]$ ▷ combination of real and initial decoy data
2:     $\tau$                ▷ the threshold to identify bad data in state estimation
3: procedure REFINEDECOY($\mathbf{z}^{init}$)
4:     $\mathbf{z}_d = \mathbf{z}_d^{init}$
5:     $[\mathbf{r} \;\; \mathbf{r}_d]$ = SE($\mathbf{z}^{init}$) = SE($[\mathbf{z} \;\; \mathbf{z}_d^{init}]$)    ▷ state estimation $SE$ returns measurement errors $\mathbf{r}$ and $\mathbf{r}_d$ for real and decoy data
6:     while $||[\mathbf{r} \;\; \mathbf{r}_d]|| > \tau$ do
7:         $\mathbf{z}_d = \mathbf{z}_d - \mathbf{r}_d$
8:         $[\mathbf{r} \;\; \mathbf{r}_d]$ = SE($[\mathbf{z} \;\; \mathbf{z}_d]$)
       return $\mathbf{z}_d$
```

In future smart grids, we can also experience increased data acquisition frequency. For example, phasormeasurement units can collect data at up to 200 times per second [7]. Crafting decoy data in such frequency can be challenging. However, those technical advancements are faced by both existing state estimation application and adversaries, who may down-sample data to prepare attacks. Consequently, instead of catching up with advanced data acquisitions, DefRec needs to compete with adversaries or existing state estimation application, crafting decoy data at a pace quicker than the pace of adversaries' preparations. Also, based on our evaluation in Figure 17, the latency of decoy data construction, built on top of state estimation, is on the same order of magnitude as the latency of state estimation. As future smart grids introduce advanced state estimation algorithms, we can increase the efficiency of decoy data construction correspondingly.

VI. IMPLEMENTATION

To evaluate security policies included in DefRec, we implemented PFV as an SDN application in the ONOS network operating system and developed a testbed (shown in Figure 7) that simulates both cyber and physical infrastructures of power grids.

Communication Networks. The network implementation follows typical setups for SDN evaluations [78]. Specifically, we used five HP ProCurve 3500yl switches and seven HP ProLiant DL3600 servers. Each switch has 48 ports, and we extended each server by deploying four PCI 4-port Ethernet adapters [26]. By grouping switch ports into different VLANs, we built six networks of different sizes (up to 124 nodes) from TopologyZoo dataset [34], which includes topology of real networks managed by different Internet Service Providers (ISPs) (see Table I).

Implementation of PFV & DefRec. PFV does not require a dedicated virtual environment; we implemented PFV as an SDN application in ONOS [4], including around 1,500 lines of code (LOC). We present storage overhead of PFV in Appendix C. Based on PFV, we also implemented DefRec's disruption policy in ONOS, using less than 200 LOC. For DefRec's attack-misleading policy, we implemented decoy data construction algorithm by using MATPOWER, an open-source MATLAB toolbox [79]; the implementation uses around 400 LOC. All implementations were carried out on a 64-bit Ubuntu 18.04, deployed in a workstation with four Intel Xeon 2.8 GHz processors and 16 GB RAM.

Physical Devices (Seed Devices). We used IEDs from three different vendors as end devices: Schweitzer Engineering Laboratories (SEL) 751A feeder protection relay [60], Allen Bradley (AB) MicroLogix 1400 PLC [1], and Schneider Electric (SE) ION7550 power meters [58]. To communicate with those devices, we implemented a DNP3 master by using the openDNP3 library [49]. For each evaluation case, we used an DNP3 master to issue requests, including data acquisition retrieving analog data and control operations opening/closing breakers, to both real devices and virtual nodes.

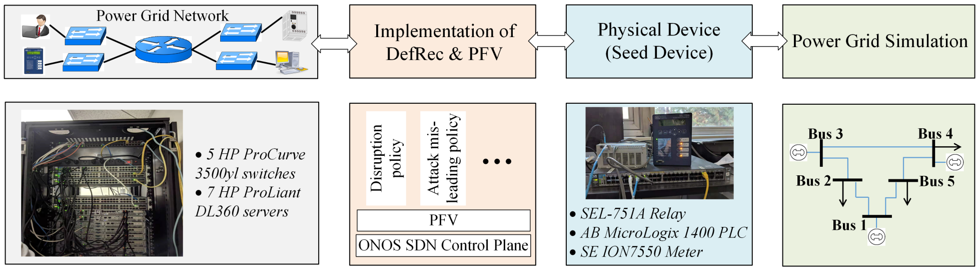


Fig. 7: **Cyber-physical testbed for evaluation.** We set the delay of network links to a normal distribution with 20 milliseconds (ms) mean and 5 ms jitter, adjusting settings of general-purpose wide-area networks [31] to meet the requirements of power grids [28].

TABLE I: **Evaluation cases.** We include the number of nodes for each network in parentheses.

Case	***Power Grid Simulation***	***Network* (# of nodes)**
1	IEEE 24-bus	Datax (11)
2	IEEE 30-bus	Abilene (22)
3	RTS96 73-bus	Hurricane (30)
4	IEEE 118-bus	Chinanet (56)
5	Poland 406-bus	Cesnet (78)
6	Poland 1153-bus	Forthnet (124)

Power Grid Simulations. To provide physical data for network traffic, we used MATPOWER to simulate six power systems [79], shown in Table I. The latter two systems represent the biggest two areas of Polish 400-, 220-, and 110-kV national transmission networks. To simulate the normal variation of operational data, we developed a benchmark profile based on data from real utilities [16], [52]. We extracted one month of real data on power generation and calculated the ratio between actual data value at each timestamp to the peak value of that month. For each simulated system, we randomly selected power generators and load units, adjusting measurements for each unit according to the benchmark.

VII. Evaluation

In this section, we perform security and performance evaluation for both PFV and security policies in DefRec.

A. *Security Evaluation*

Security evaluation focuses on the effectiveness of: (i) PFV's virtualization on network flows of real devices, (ii) disruption policy to delay passive attacks and isolate proactive attacks (RO1 and RO2 in Section II-B), and (iii) attack-misleading policy causing adversaries to design ineffective attacks (RO3).

1) Effectiveness of PFV's Virtualization: We evaluate PFV's effectiveness on actual implementation of network stacks, system invariants, and physical state variations of real devices.

Evaluation of Network Stack Implementation. We used three outputs from experiments to verify this implementation. *First*, outputs in the DNP3 master triggered by responses from real devices and virtual nodes were always consistent. *Second*, network packets showed no errors in common network analysis tools, such as Wireshark and Zeek runtime network analyzer [50], [73]. *Last*, SDN controllers recorded lower-layer network information of all virtual nodes, e.g., their entries in ARP caches and corresponding flow entries.

Evaluation of System Invariants. We applied fingerprinting methods proposed for ICSs on both real physical devices and virtual nodes. As shown in [18], the time to execute commands in ICS devices, e.g., data acquisition and control commands, is an effective system invariant to identify device types and models. Based on the methods presented in [18], we measured the difference between the timestamp in the response carrying measurement data and the timestamp in the corresponding TCP acknowledgment, to accurately reveal execution times in real devices or virtual nodes.

In Figure 8, we show the estimated execution times based on responses from three IEDs and from corresponding virtual nodes. Three IEDs show different characteristics if we consider the combination of average and variations of execution times. For example, SE ION 7550 has the largest execution time and also the biggest variation. The execution times of SEL 751A and AB MicroLogix 1400 have a close average but different variations. Outbound packets of virtual nodes, tailored based on profiles of those real devices, follow the communication patterns, making it challenging for adversaries to distinguish between real devices and virtual nodes.

In Figure 9, we show the probability density functions (PDFs) of execution time measured for both data acquisition and control operations. We can see that PDF patterns vary in different operation types and devices. In all cases, virtual nodes can follow the communication patterns of real devices. We only observe minor differences in the execution time between them, less than 2 ms, which falls within normal variations caused by factors like locations and configurations of devices and switches.

Evaluation of Physical State Variations. Power grids use voltages at different locations to represent physical states. Variation of these values is a critical metric of health condition of a power grid. In Figure 10, we present the voltage magnitude measured by the real devices and by the corresponding virtual nodes, which is normalized to a reference substation (i.e., a "slack bus"). Based on device profiles, virtual nodes can follow physical state variations of real devices, with less than 1% differences. Physical states of virtual nodes are not exactly the same as ones of real devices, which are adjusted by the packet hooking component according to device profiles.

2) Effectiveness of the Disruption Policy: The disruption policy included in DefRec is to achieve RO1 and RO2, i.e., significantly delay *passive* and *proactive* attacks.

Effectiveness in RO1. We achieve RO1 by randomizing network requests issued to both real devices and *accessible* virtual nodes. Based on the analysis in Section IV, we estimate the time for adversaries to obtain the identities of real devices

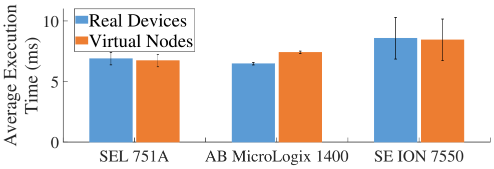
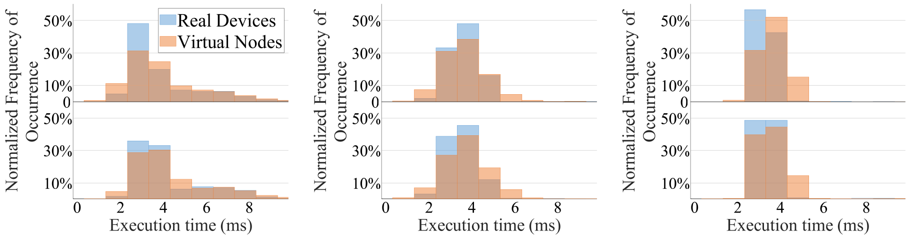


Fig. 8: **Comparing the execution time** (with 99% confidence interval) in physical devices and virtual nodes.



(a) SEL 751A

(b) AB MicroLogix 1400

(c) SE ION 7550

Fig. 9: **PDF** (y-axis) **of execution time** (x-axis) of data acquisition (at top) and control operations (at bottom) for three IEDs.

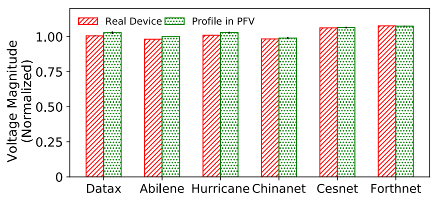


Fig. 10: **Voltage magnitude** measured by real devices and profiled in PFV (with 99% confidence interval).

against the number of *accessible* virtual nodes. We make such estimation by assuming that an adversary can passively monitor up to 200 network packets every second (the most frequent data acquisition that can be observed in modern power grids [7]).

As shown in Figure 11, for each power grid, we present the *accessible*-virtual-node to real-device ratio (shown in y-axis) that is sufficient to delay adversaries' inference of the identities of real devices for at least 100, 1000, 10,000, and 100,000 years (illustrated in four plots). Even for a small 24-bus power grid, we can delay passive attacks for 100 years by adding a number of accessible virtual nodes equal to 15% of the number of real devices. As the size of a power grid increases, the ratio decreases exponentially. This is because the probability that adversaries will make a correct guess about a real device decreases significantly with the size of a power grid (see Figure 19 in Appendix A).

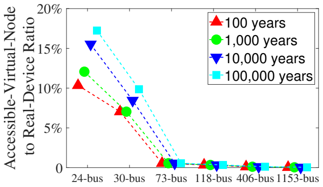


Fig. 11: ***Accessible*-virtual-node to real-device ratio** sufficient to delay passive attacks.

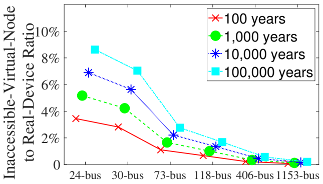


Fig. 12: ***Inaccessible*-virtual-node to real-device ratio** sufficient to delay proactive attacks.

Effectiveness in RO2. Similar to passive IDSs detecting malicious activities, we redefine false positive and negative (FP and FN), to quantify the effectiveness of the disruption policy in identifying and isolating proactive attacks.

Definition 3. RO2 FN: adversaries successfully obtain the measurements from real devices.

Definition 4. RO2 FP: legitimate applications access *inaccessible* virtual nodes. Note that there are no FPs for properly-configured applications that know the identities of real devices. In Appendix A, we present FPs for applications that are misconfigured or unaware of DefRec.

The proposed probabilistic dropping protocol makes FN rate of RO2 very low, e.g., less than 10^{-10} even for a small 24-bus power grid (see Figure 21 in Appendix A). To better interpret those results, we estimate the time for adversaries to learn real measurements through *proactive* attacks against the *inaccessible*-virtual-node to real-device ratio. Similar to the evaluation of RO1, we assume that adversaries can actively probe control networks with a throughput of 10 Gigabytes per second. In practice, probing a network with such or bigger throughput can easily trigger alerts in many IDSs, such as Zeek [50].

In Figure 12, we present the *inaccessible*-virtual-node to real-device ratio (shown in y-axis) that is sufficient to delay adversaries' acquisition of real measurements for at least 100, 1000, 10,000, and 100,000 years. Even for a small 24-bus power grid, we need to add a number of inaccessible virtual nodes equal to 4% of the number of real devices to achieve RO2. Consequently, when the probabilistic dropping protocol is used, adversaries must generate a large number of probes to identify real measurements. If they use the combination of real and decoy measurements to prepare attacks, the attack-misleading policy (evaluated in Section VII-A3) can cause adversaries to design ineffective strategies.

Discussion. Figure 11 and Figure 12 show that DefRec can delay adversaries for a long latency. In that span of time, adversaries can face other challenges. For example, operational environments of power grids can experience changes with deployment of new generators (see Figure 23 in Appendix A) to satisfy increasing load demands [27], resulting in changes of network and physical configurations. Those changes can make the previously obtained knowledge of a power grid obsolete, further increasing adversaries' difficulty in reconnaissance.

3) Effectiveness of the Attack-Misleading Policy: When adversaries implement ***active*** attacks on FDIAs based on real data, vulnerable state estimation introduces no alerts.

When adversaries implement FDIAs based on decoy data (in responses from *inaccessible* and/or *accessible* virtual nodes), misled attacks trigger alerts in state estimation. In this section, we quantify the effectiveness of decoy data and confirm the theoretical findings. Based on the evaluations of RO1 and RO2, we change the virtual-node to real-device ratio from 0 to 20%. Also, we change the ratio of compromised measurements (i.e., the ratio of non-zero entries in vector $\mathbf{a}$; see Section V-A for its definition) from 0 to 100%, quantifying adversaries' capability in *active* attacks. As discussed in [42], more compromised measurements lead to a higher probability of successful FDIAs, but can become easier to detect.

Definition 5. RO3 FN: FDIAs prepared based on decoy data are successful. We consider an FDIA as successful if compromised measurements determined based on decoy data introduce no alerts in state estimation, i.e., L_2-norm of measurement residual satisfies the condition $||\mathbf{z}'_a - H\hat{\mathbf{x}}'_a|| \leq \tau$.

Definition 6. RO3 FP: decoy data are not valid, meaning that the combination of decoy and real data raises alerts in state estimation without FDIAs. FPs of decoy data do not increase

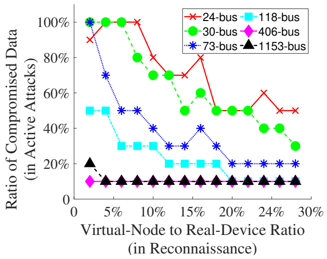


Fig. 13: **Virtual nodes vs. compromised data** when FN rate is less or equal to 0.5%.

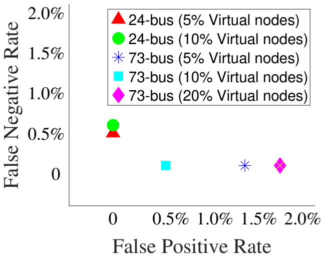


Fig. 14: **FP/FN rates of decoy data.**

the overhead of state estimation, but require DefRec to craft a different set of decoy data.

In Figure 13, we show that decoy data can restrict adversaries' capability in *active* attacks. For each amount of decoy data in the *x*-axis (measured by the virtual-node to real-device ratio), we mark a ratio of compromised measurements used in successful *active* attacks, above which FN rates become less than or equal to 0.5%. Based on the results, we can see that even if adversaries become very careful and compromise less than 10% of measurements in their *active* attacks on a normal scale power grid, the attack-misleading policy can still expose their malicious activities in state estimation.

In Figure 14, we combine FP and FN rates of decoy data. Because there were no FPs and FNs in most evaluation cases, we only present the case when either FP or FN rate is not zero. For the small 24-bus system, we observe less than 0.5% FN rate. We found that even in an FN event, Equation (6) did not hold, but residual errors were accidentally small, especially when a small ratio of measurements were compromised. As the size of a power grid and the number of decoy data increase, residual errors increase dramatically, at least 5,000 times of detection threshold or $||\mathbf{z}'_a - H\hat{\mathbf{x}}'_a|| \geq 5000\tau$ (see Figure 22 in Appendix A).

As shown in the figure, we only found FP for the 73-bus RTS96 system with less than 2% occurrences. This is mainly because physical components of this system are closely correlated (e.g., with some transmission lines delivering a large amount of active/reactive power). Consequently, as we add virtual nodes, adjusting the value of decoy data can have comparatively more impact on real devices than other systems.

B. Performance Evaluation

According to the security evaluations in Section VII-A, it is sufficient to achieve RO1 and RO2 by varying the ratio of accessible and inaccessible virtual nodes to real devices from 5% to 20%. Consequently, we focus on performance evaluation by varying the ratio of virtual nodes in this range. The *accessible* virtual nodes are periodically accessed by randomized requests, consuming network bandwidth. The *inaccessible* virtual nodes introduce minimal runtime overhead, as DefRec isolates proactive attacks after a few attempts.

1) The Capability of PFV: Two factors affect PFV's performance: the capability of virtualizing physical devices and overhead of device profiles and caching network interactions.

Overhead of Packet Hooking. To quantify PFV's capability to hook network packets, we measured the goodput of the SDN controller application implementing the packet hooking component. In Figure 15, we show the average goodput (in Megabits per second, Mbps) with 99% confidence interval. We use the *x*-axis to separate results of six evaluation cases and the ratio of virtual nodes.

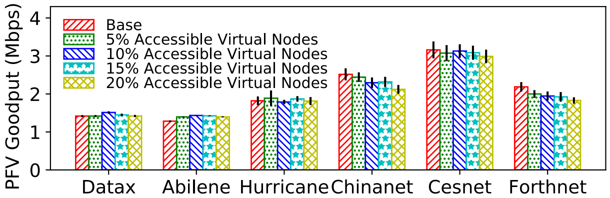


Fig. 15: **Capability of packet hooking.** The goodput is between 1.5 and 3.0 Mbps.

The results show that the goodput is at least 1.5 Mbps and does not vary significantly with the ratio of virtual nodes. PFV's performance benefits from the fact that its SDN application performs simple tasks, compared to SDN controllers that perform complicated tasks in general-purpose networks, such as determining network topology and identifying machine locations. For a DNP3 packet of 256 bytes, which can contain more than 64 32-bit measurements, PFV can process around 600 packets per second, which is equivalent to processing 30,000 decoy data on a single site.

Overhead of Device Profiles & Caching. The storage overhead of device profiles is closely related to the number of physical state of a power grid and the number of virtual nodes, while caching overhead is related to the types of network requests and responses used by power grids. In Table II and Table III in Appendix C, we present the estimated storage overhead based on power system cases used in evaluations. We expect to occupy around 370 KB to profile data of a large power grid and 10 KB to cache network interactions with real devices.

2) Impact of the Disruption Policy: The disruption policy specified in DefRec introduces additional network traffic from virtual nodes. We conducted experiments to understand the impact of the injected packets on the performance of existing networks.

We measured and compared round-trip time (RTT) of all data acquisitions and control operations with and without DefRec enabled. In Figure 16, we show average RTT (with 99% confidence interval) in milliseconds; we group results by different evaluation cases, within which we use different bar patterns to represent the ratio of virtual nodes.

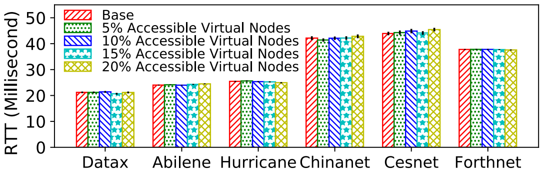


Fig. 16: **Comparing RTT with and without DefRec enabled.** The *x*-axis specifies evaluation cases and the ratio of virtual nodes; the *y*-axis indicates RTT in milliseconds (ms).

As shown in Figure 16, we observed a negligible impact on the RTT of normal communication. For each case, variations are within ±3%, which is on the same order of magnitude as normal jitters that we can observe in communication networks.

Also because of those jitters, we found that average RTTs are slightly smaller when DefRec randomizes network packets in some networks. The results also suggest that RTT is not only affected by the size of a network but also by its topology. For example, even though "Cesnet" network has fewer nodes than "Forthnet" network, the former has more paths with a longer latency than the latter, which needs more time to deliver data.

3) Performance of the Attack-Misleading Policy: The attack-misleading policy crafts decoy data; a long execution time to craft them can make it less feasible to deploy DefRec in power grids. In Figure 17, we show the execution time to craft 20% decoy data (with 99% confidence interval that is hardly noticeable). While constructing decoy data takes less than 0.3 seconds for most power systems, it takes around 4.7 seconds for Poland 1153-bus system. In practice, we determine a new set of decoy data when real data in a site experience significant changes. Because of mechanical inertia in many power grids and other ICSs, real data usually change slowly, e.g., on the order of minutes [54]. Therefore, the execution time is acceptable in practice for deploying DefRec in a real utility environment.

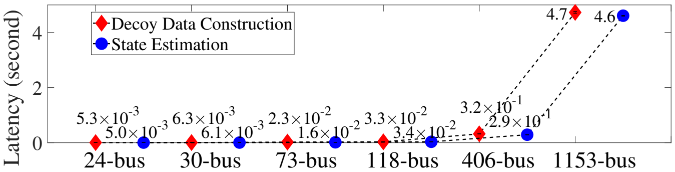


Fig. 17: **The execution time to craft decoy data.** We can craft the decoy data within 5 seconds for large-scale Poland 1,153-bus system.

In Figure 17, we also compare the result with the execution time of state estimation performed on the same power system. Because we construct decoy data by using components in state estimation (see Algorithm 1), the time to craft decoy data is on the same order of magnitude as the execution time of state estimation. In future smart grids, when state estimation algorithm evolves, we can expect that decoy data construction can take less time.

VIII. Related Work

Network Function Virtualization (NFV). NFV is an emerging technology to virtualize network nodes according to specific functionality, such as load balancing and access control [48]. NFV is not necessarily dependent on SDN, but SDN's network programmability and visibility can significantly benefit its design. Recent work has applied NFV to improve performance and flexibility of security designs. Li et al. propose to use NFV to virtualize detection logic of network IDSs, allowing efficient and flexible state sharing and resource migration. Deng et al. leverage SDN and NFV to overcome resource limitations of hardware-based firewall applications, enabling elastic and scalable access control for virtual computing environments. Inspired by NFV, PFV aims at virtualizing physical devices by using SDN to hook network packets from them and to tailor the packets with intelligently crafted decoy data. By following actual behavior of real devices, PFV can significantly disrupt passive and proactive attacks.

Moving Target Defense (MTD) in ICSs. Traditional MTD approaches disrupt adversaries by randomly changing system and network configurations, e.g., IP addresses and port numbers [3], [30], [33], [76]. Some recent work leverages similar designs to disrupt control operations in ICSs. Rahman et al. randomly change the set of physical data used for power system analysis, attempting to remove some compromised data and to reduce the effectiveness of FDIAs [51]. Another group of MTD approaches intentionally disrupt physical process in an ICS and use deviations from expected consequences to detect attacks [2], [14], [46], [71]. Those approaches require physical perturbations, which can harm the existing physical process. In PFV, we require no modification on existing cyber and physical infrastructures of a power grid; security policies included in DefRec are preemptive, not passive, disrupting reconnaissance before malicious activities occur.

Among current MTD approaches, RAINCOAT is the most similar one to our approach, as it also manipulates network packets to disrupt adversaries' reconnaissance in power grids [40]. However, RAINCOAT spoofs network packets from existing physical devices and delivers both spoofed data and real data from the same device in a time-sharing manner. This approach can significantly increase the amount of network traffic by at least 50%. DefRec, on the other hand, relies on a small amount of decoy data from virtual nodes to disrupt reconnaissance of both cyber and physical infrastructures, significantly reducing interference in real devices.

Honeypots for ICSs. Honeypots or honeynets interact with adversaries with simulated network packets [56], [68]. Several honeypot projects aim at building separate computing or network environments to trace adversaries' activities on ICS devices, e.g., PLCs [6], [11], [72]. Han et al. further propose to use SDN to automate interactions with adversaries [22]. Those ICS honeypots can mimic a cyberinfrastructure of an ICS. However, in their constructed networks, the honeypots lack supports for constructing meaningful application-layer payloads, e.g., measurements exchanged between ICS devices.

Instead of mimicking and simulating network packets, we design PFV, a completely different technique, by virtualizing physical devices. PFV is not a honeypot for ICSs: it does not require interactions with adversaries to disrupt their reconnaissance. Adversaries that passively monitor network packets can be significantly delayed; they can end up using decoy data to design damage-free attack strategies.

Masquerading Attacks in Remote Attestations. Remote attestation is a technique used by a device (verifier) to verify properties of another remote device (prover), such as its software integrity [53], policy enforcement [67], or physical locations [13]. Remote attestation can be vulnerable to masquerading or relay attacks, in which a malicious prover forwards requests from a verifier to another legitimate device (victim) and then use the responses from this victim device as a proof [44]. To disable masquerading attacks, it is necessary to include device-specific information in the response, such that the verifier can distinguish between responses from the victim and the ones from the prover. This device-specific information can be secret keys for a TPM (Trusted Platform Module) [67] or round trip times used in distance bounding protocols [13], [43].

In PFV, the packet hooking component forwards requests destined to virtual nodes to real devices; this procedure is similar to masquerading attacks. However, in PFV, virtual nodes and real devices are working collaboratively, e.g., allowing virtual nodes to cache or tailor packets sent from real devices. Also, communication channels between them can be encrypted to avoid adversaries' eavesdropping. Consequently, it is challenging for adversaries, playing a similar role as a verifier, to use existing protection mechanisms for masquerading attacks to distinguish between responses from virtual nodes and ones from real devices.

Increasing SDN's resilience. Rich capabilities provided by SDN also make SDN controllers a popular target of attacks [24], [57], [63], [69]; therefore, SDN-based approaches require more complex protections. DefRec makes a very light use of SDN's network programmability, i.e., hooking a small number of packets (this can be implemented by other network manipulation techniques). Monitoring DefRec's behavior and verifying its integrity is efficient, compared to verifying full-fledged SDN controllers.

IX. Conclusion

In this paper, we propose the concept of PFV, which hooks network interactions with real devices to build virtual nodes. Lightweight virtual nodes built by PFV follow actual implementations of network stacks, system invariants, and physical state variations of real devices. Based on PFV, DefRec specifies two security policies, randomizing communications and crafting decoy data for virtual nodes, to disrupt adversaries' reconnaissance of power grids' cyber-physical infrastructures. Based on evaluations on real devices and large-scale power grids, we find that DefRec can successfully delay passive and proactive attacks for 100 years with a small number of virtual nodes, and successfully mislead adversaries into designing ineffective attacks. In addition, DefRec introduces negligible overhead on existing network communications (e.g., less than 3% on RTTs) and control operations.

In future work, we will provide formal coverage analysis of PFV's functionalities and apply it into more ICS environments, including parsers and encoders for protocols used in different ICS networks. Also, we will extend PFV with other IDS techniques to evaluate its performance on attacks requiring no reconnaissance or targeting on data privacy.

Acknowledgment

This material is based upon work partially supported by the National Science Foundation under Award No. CNS-1850377 and the National Science Foundation under Award No. CNS-1717313. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] "Allen Bradley MicroLogix 1400 programmable logic controller systems," [Online] Available at: https://ab.rockwellautomation.com/Programmable-Controllers/MicroLogix-1400.

[2] M. Q. Ali and E. Al-Shaer, "Randomization-based intrusion detection system for advanced metering infrastructure," *ACM Transactions on Information and System Security*, vol. 18, no. 2, pp. 1–30, 2015.

[3] S. Antonatos, P. Akritidis, E. Markatos, and K. Anagnostakis, "Defending against hitlist worms using network address space randomization," *Computer Networks*, vol. 51, no. 12, pp. 3471–3490, 2007.

[4] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and others, "ONOS: towards an open, distributed SDN OS," in *Proceedings of the 3rd workshop on Hot topics in software defined networking*. ACM, 2014, pp. 1–6.

[5] R. B. Bobba, K. M. Rogers, Q. Wang, H. Khurana, K. Nahrstedt, and T. J. Overbye, "Detecting false data injection attacks on DC state estimation," in *Preprints of the First Workshop on Secure Control Systems, CPSWEEK*, 2010, pp. 18–26.

[6] D. I. Buza, F. Juhász, G. Miru, M. Félegyházi, and T. Holczer, "CryPLH: Protecting smart energy systems from targeted attacks with a PLC honeypot," in *Smart Grid Security*, J. Cuellar, Ed. Springer International Publishing, 2014, pp. 181–192.

[7] "IEEE Standard for Synchrophasor Data Transfer for Power Systems," pp. 1–53, Dec 2011.

[8] A. A. Cardenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: Risk assessment, detection, and response," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (ASIACCS '11)*, 2011, pp. 355–366.

[9] Chi-Ho Tsang and Sam Kwong, "Multi-agent intrusion detection system in industrial network using ant colony clustering approach and unsupervised feature extraction," in *2005 IEEE International Conference on Industrial Technology*, 2005, pp. 51–56.

[10] E. Chien, L. OMurchu, and N. Falliere, "W32.duqu: The precursor to the next stuxnet," in *5th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 12)*. San Jose, CA: USENIX Association, Apr. 2012.

[11] "Conpot: low interactive server side industrial control systems honeypot," [Online] Available at: http://conpot.org/.

[12] V. Costan and S. Devadas, "Intel sgx explained." *IACR Cryptology ePrint Archive*, vol. 2016, no. 086, pp. 1–118, 2016.

[13] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, "Distance hijacking attacks on distance bounding protocols," in *2012 IEEE Symposium on Security and Privacy*, May 2012, pp. 113–127.

[14] K. R. Davis, K. L. Morrow, R. Bobba, and E. Heine, "Power flow cyber attacks and perturbation-based defense," in *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, Nov 2012, pp. 342–347.

[15] M. Dhawan, R. Poddar, K. Mahajan, and V. Mann, "SPHINX: Detecting security attacks in software-defined networks," in *Proceedings 2015 Network and Distributed System Security Symposium*. Internet Society, 2015.

[16] "U.S. energy information administration - EIA - independent statistics and analysis," [Online] Available at: https://www.eia.gov/opendata/.

[17] N. Falliere, L. O. Murchu, and E. Chien, "W32. stuxnet dossier," *White paper, Symantec Corp., Security Response*, vol. 5, no. 6, p. 29, 2011.

[18] D. Formby, P. Srinivasan, A. Leonard, J. Rogers, and R. Beyah, "Who's in control of your control system? device fingerprinting for cyber-physical systems," in *Proceedings 2016 Network and Distributed System Security Symposium*. Internet Society, 2016.

[19] L. A. Garcia, F. Brasser, M. H. Cintuglu, A.-R. Sadeghi, O. Mohammed, and S. A. Zonouz, "Hey, my malware knows physics! attacking PLCs with physical model aware rootkit," in *Proceedings 2017 Network and Distributed System Security Symposium*. Internet Society, 2017.

[20] J. D. Glover, M. S. Sarma, and T. Overbye, *Power System Analysis & Design, SI Version*. Cengage Learning, 2012.

[21] A. Goodney, S. Kumar, A. Ravi, and Y. H. Cho, "Efficient PMU networking with software defined networks," in *2013 IEEE International Conference on Smart Grid Communications (SmartGridComm)*. IEEE, 2013, pp. 378–383.

[22] W. Han, Z. Zhao, A. Doupé, and G.-J. Ahn, "HoneyMix: Toward SDN-based intelligent honeynet," in *Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization - SDN-NFV Security '16*. ACM Press, 2016, pp. 1–6.

[23] C. Hoga and G. Wong, "IEC 61850: open communication in practice in substations," in *IEEE PES Power Systems Conference and Exposition*, vol. 2, Oct 2004, pp. 618–623.

[24] S. Hong, L. Xu, H. Wang, and G. Gu, "Poisoning network visibility in software-defined networks: New attacks and countermeasures," in *Proceedings 2015 Network and Distributed System Security Symposium*. Internet Society, 2015.

[25] A. Houmansadr, C. Brubaker, and V. Shmatikov, "The parrot is dead: Observing unobservable network communications," in *2013 IEEE Symposium on Security and Privacy*, May 2013, pp. 65–79.

[26] "HP Ethernet 1Gb 4-port 331FLR Adapter - Specifications," 2012, [Online] Available at: https://support.hpe.com/hpsc/doc/public/display?docLocale=en_US&docId=emr_na-c03226012.

[27] J. Huang, C. Jiang, and R. Xu, "A review on distributed energy resources and microgrid," *Renewable and Sustainable Energy Reviews*, vol. 12, no. 9, pp. 2472 – 2483, 2008.

[28] "IEEE standard communication delivery time performance requirements for electric power substation automation," 2005, [Online] Available at https://standards.ieee.org/standard/1646-2004.html.

[29] "IEEE standard for electric power systems communications-distributed network protocol (dnp3)," pp. 1–821, Oct 2012, [Online] Available at https://standards.ieee.org/standard/1815-2012.html.

[30] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "An effective address mutation approach for disrupting reconnaissance attacks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2562–2577, Dec 2015.

[31] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined wan," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, New York, NY, USA, 2013, pp. 3–14.

[32] F. Kelbert, F. Gregor, R. Pires, S. Köpsell, M. Pasin, A. Havet, V. Schiavoni, P. Felber, C. Fetzer, and P. Pietzuch, "Securecloud: Secure big data processing in untrusted clouds," in *Proceedings of the Conference on Design, Automation & Test in Europe*, ser. DATE '17. 3001 Leuven, Belgium, Belgium: European Design and Automation Association, 2017, pp. 282–285.

[33] D. Kewley, R. Fink, J. Lowry, and M. Dean, "Dynamic approaches to thwart adversary intelligence gathering," in *Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01*, vol. 1. IEEE Comput. Soc, 2001, pp. 176–185.

[34] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

[35] O. Kosut, L. Jia, R. J. Thomas, and L. Tong, "Malicious data attacks on the smart grid," *IEEE Transactions on Smart Grid*, vol. 2, no. 4, pp. 645–658, 2011.

[36] R. Lasseter, A. Akhil, C. Marnay, J. Stephens, J. Dagle, R. Guttromsom, A. S. Meliopoulous, R. Yinger, and J. Eto, "Integration of distributed energy resources: the certs microgrid concept," Consortium Electric Reliability Technology Solution, Tech. Rep., October 2003.

[37] D. Lee, "Ukraine power cut was cyber-attack," Jan 2017, [Online] Available at: https://www.bbc.com/news/technology-38573074.

[38] R. M. Lee, M. J. Assante, and T. Conway, "Analysis of the cyber attack on the ukrainian power grid," SANS and E-ISAC, Tech. Rep., March 2016.

[39] H. Lin, H. Alemzadeh, D. Chen, Z. Kalbarczyk, and R. K. Iyer, "Safety-critical cyber-physical attacks: Analysis, detection, and mitigation," in *Proceedings of the Symposium and Bootcamp on the Science of Security (HotSoS)*. ACM, 2016, pp. 82–89.

[40] H. Lin, A. Slagell, Z. Kalbarczyk, and R. K. Iyer, "Raincoat: Randomization of network communication in power grid cyber infrastructure to mislead attackers," *IEEE Transactions on Smart Grid*, pp. 1–1, 2018.

[41] H. Lin, A. Slagell, Z. Kalbarczyk, P. Sauer, and R. K. Iyer, "Runtime semantic security analysis to detect and mitigate control-related attacks in power grids," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 163–178, Jan 2018.

[42] Y. Liu, P. Ning, and M. K. Reiter, "False data injection attacks against state estimation in electric power grids," *ACM Transactions on Information and System Security*, vol. 14, no. 1, pp. 1–33, 2011.

[43] S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua, "Distance-bounding protocols: Verification without time and location," in *2018 IEEE Symposium on Security and Privacy (SP)*, May 2018, pp. 549–566.

[44] J. M. McCune, A. Perrig, A. Seshadri, and L. van Doorn, "Turtles all the way down: Research challenges in user-based attestation," in *USENIX Summit on Hot Topics in Security (HotSec '07)*. USENIX Association, 2007.

[45] A. Monticelli, "Electric power system state estimation," *Proceedings of the IEEE*, vol. 88, no. 2, pp. 262–282, 2000-02. [Online]. Available: http://ieeexplore.ieee.org/document/824004/

[46] K. L. Morrow, E. Heine, K. M. Rogers, R. B. Bobba, and T. J. Overbye, "Topology perturbation for detecting malicious data injection," in *2012 45th Hawaii International Conference on System Sciences*, 2012, pp. 2104–2113.

[47] K. G. Nagananda, S. Kishore, and R. S. Blum, "A PMU scheduling scheme for transmission of synchrophasor data in electric power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2519–2528, 2015-09.

[48] "Network functions virtualisation: An introduction, benefits, enablers, challenges, and call for action," Oct 2012, [Online] Available at: https://portal.etsi.org/nfv/nfv_white_paper.pdf.

[49] "Opendnp3: the de facto reference implementation of ieee-1815 (dnp3)," [Online] Available at: https://www.automatak.com/opendnp3/.

[50] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer Networks*, p. 29, 1999.

[51] M. A. Rahman, E. Al-Shaer, and R. B. Bobba, "Moving target defense for hardening the security of the power system state estimation," in *Proceedings of the First ACM Workshop on Moving Target Defense - MTD '14*. ACM Press, 2014, pp. 59–68.

[52] "PJM: regional transmission organization (rto) that coordinates the movement of wholesale electricity in all or parts of 13 states and the district of columbia," [Online] Available at: https://www.pjm.com/.

[53] R. Sailer, X. Zhang, T. Jaeger, and L. Van Doorn, "Design and implementation of a tcg-based integrity measurement architecture." in *13th USENIX Security Symposium (USENIX Security '04), publisher = USENIX Association*, 2004, pp. 223–238.

[54] P. W. Sauer and M. A. Pai, *Power system dynamics and stability*. Prentice Hall Upper Saddle River, NJ, 1998, vol. 101.

[55] G. W. Scheer and D. J. Dolezilek, "Comparing the reliability of ethernet network topologies in substation control and monitoring networks," in *Western Power Delivery Automation Conference, Spokane, Washington*, 2000.

[56] M. Schloesser, "Dionaea low interaction honeypot (forked from dionaea.carnivore.it): rep/dionaea," [Online] Available at: https://github.com/rep/dionaea.

[57] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN security: A survey," in *2013 IEEE SDN for Future Networks and Services (SDN4FNS)*. IEEE, 2013-11, pp. 1–7.

[58] "Schneider Electric PowerLogic ion7550/ion7650 series," [Online] Available at: https://www.schneider-electric.us/en/product-range/1460-powerlogic-ion7550-ion7650-series/.

[59] "SEL-2740s software-defined network switch," [Online] Available at: https://selinc.com/products/2740S/.

[60] "SEL-751a feeder protection relay," [Online] Available at: https://selinc.com/products/751A/.

[61] D. Shelar, P. Sun, S. Amin, and S. Zonouz, "Compromising security of economic dispatch in power system operations," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 6 2017, pp. 531–542.

[62] L. V. Silva, R. Marinho, J. L. Vivas, and A. Brito, "Security and privacy preserving data aggregation in cloud computing," in *Proceedings of the Symposium on Applied Computing*, ser. SAC '17. New York, NY, USA: ACM, 2017, pp. 1732–1738.

[63] R. Skowyra, L. Xu, G. Gu, V. Dedhia, T. Hobson, H. Okhravi, and J. Landry, "Effective topology tampering attacks and defenses in software-defined networks," in *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2018-06, pp. 374–385.

[64] S. Soltan, P. Mittal, and H. V. Poor, "BlackIoT: IoT botnet of high wattage devices can disrupt the power grid," in *27th USENIX Security Symposium*. USENIX Association, 2018, pp. 15–32.

[65] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *2010 IEEE Symposium on Security and Privacy*. IEEE, 2010, pp. 305–316.

[66] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with rocketfuel," in *Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '02. New York, NY, USA: ACM, 2002, pp. 133–145.

[67] F. Stumpf, O. Tafreschi, P. Röder, and C. Eckert, "A robust integrity reporting protocol for remote attestation," Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL), Publications of Darmstadt Technical University, Institute for Business Studies (BWL), 2006.

[68] U. Tamminen, "Kippo: SSH honeypot. contribute to desaster/kippo development by creating an account on GitHub," 2018-12-02, [Online] Available at: https://github.com/desaster/kippo.

[69] B. E. Ujcich, U. Thakore, and W. H. Sanders, "Attain: An attack injection framework for software-defined networking," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2017, pp. 567–578.

[70] X. Wang, C. Xu, K. Wang, F. Yan, and D. Zhao, "Toward cost-effective memory scaling in clouds: Symbiosis of virtual and physical memory," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, vol. 00, 2018-07, pp. 33–40.

[71] S. Weerakkody, Y. Mo, and B. Sinopoli, "Detecting integrity attacks on control systems using robust physical watermarking," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 3757–3764.

[72] K. Wilhoit and S. Hilt, "The GasPot experiment: Unexamined perils in using gas tank monitoring systems," A TrendLabs Research Paper, Trend Micro Inc. [Online]. Available: https://www.trendmicro.de/media/wp/the-gaspot-experiment-wp-en.pdf

[73] "Wireshark go deep," [Online] Available at: https://www.wireshark.org/.

[74] L. Xu, J. Huang, S. Hong, J. Zhang, and G. Gu, "Attacking the brain: Races in the SDN control plane," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 451–468.

[75] Yuan Liao and M. Kezunovic, "Online optimal transmission line parameter estimation for relaying applications," *IEEE Transactions on Power Delivery*, vol. 24, no. 1, pp. 96–102, Jan 2009.

[76] J. Yuill, D. Denning, and F. Feer, "Using deception to hide things from hackers: Processes, principles, and techniques," *Journal of Information Warfare*, vol. 3, no. 5, p. 16, 2006.

[77] M. Zalewski, *Silence on the wire: a field guide to passive reconnaissance and indirect attacks*. No Starch Press, 2005.

[78] W. Zhou, D. Jin, J. Croft, M. Caesar, and P. B. Godfrey, "Enforcing customizable consistency properties in software-defined networks," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 73–85.

[79] R. Zimmerman, C. E. Murillo-Sanchez, and R. Thomas, "MATPOWER: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12–19, 2011.

Appendix A
Additional Details on Disruption Policy

A. Details of Probabilistic Dropping Protocol

In Section IV, we propose a probabilistic dropping protocol to reduce the effectiveness of adversaries' probing activities, making successful guessing about real devices difficult. We present the protocol in Figure 18. After an adversary accesses an inaccessible virtual node (marked as device 0), we isolate the adversary when she accesses a device k with the probability p_k (including device 0), regardless of whether device k is a real device or a virtual node. We set a threshold δ such that the adversary will be isolated at her access to the device $\delta+1$. In other words, the adversary can only access at most δ real devices before it is isolated from control networks.

We model the protocol as a series of biased coin flips with increasing probabilities, i.e., $p_0, \ldots, p_k$. The probability that an adversary is isolated at her k-th access is $Q_k = p_k \times \prod_{j=0}^{k-1}(1-p_j)$ for $1 \leq k \leq \delta$.

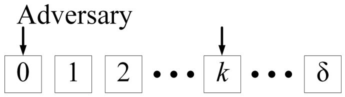


Fig. 18: **Probabilistic dropping protocol.**

Theorem 1. *The chance that an adversary will be isolated from control networks is evenly distributed, i.e.,* $p_0 = Q_0 = Q_1 = \cdots = Q_\delta$*, if* $p_k = \dfrac{p_0}{1 - k \cdot p_0}$

Proof: We prove this theorem by mathematical induction.

By definition, when $k = 0$, we have $p_0 = \dfrac{p_0}{1 - 0 \cdot p_0}$ and $Q_0 = p_0$.

Assuming that, this condition holds when $k = i$, i.e., if $p_i = \dfrac{p_0}{1 - i \cdot p_0}$, we have $Q_i = p_0$. When $k = i + 1$, we have $Q_{i+1} = p_{i+1} \times \prod_{j=0}^{i}(1 - p_j)$. Through the following derivation, we can represent Q_{i+1} in terms of Q_i:

$$\begin{aligned}
Q_{i+1} &= p_{i+1} \times \prod_{j=0}^{i}(1 - p_j) \\
&= \frac{p_{i+1}}{p_i} \times p_i \times (1 - p_i) \times \prod_{j=0}^{i-1}(1 - p_j) \\
&= \frac{p_{i+1}}{p_i} \times (1 - p_i) \times p_i \times \prod_{j=0}^{i-1}(1 - p_j) \\
&= \frac{p_{i+1}}{p_i} \times (1 - p_i) \times Q_i \qquad (7)
\end{aligned}$$

Based on the inductive step, we can have the following derivation by making $Q_{i+1} = p_0$:

$$\begin{aligned}
p_0 = Q_{i+1} &= \frac{p_{i+1}}{\dfrac{p_0}{(1 - i \cdot p_0)}} \cdot \left(1 - \frac{p_0}{1 - i \cdot p_0}\right) \cdot p_0 \Rightarrow \\
p_{i+1} &= \frac{p_0}{1 - (i+1) \cdot p_0} \qquad (8)
\end{aligned}$$

Consequently, if $p_k = \dfrac{p_0}{1 - k \cdot p_0}$, we always have $p_0 = Q_0 = Q_1 = \cdots = Q_\delta$. ■

Based on Theorem 1, we can derive the probability that an adversary successfully obtains measurements from all real devices through proactive attacks. When the adversary accesses k-th device, the probability of not being isolated is $1 - Q_k$ or $1 - p_0$. We use m_2 to represent the number of inaccessible virtual nodes and n the number of real devices. Because there are $m_2 - 1$ remaining inaccessible virtual nodes (excluding device 0 that triggers the probabilistic dropping protocol), the chance that the adversary is accessing a real device at the k-th access is $\frac{1}{\binom{n+m_2-1}{n-k+1}}$, if the previous $k-1$ devices are also real devices. Consequently, the probability that the adversary has obtained measurements from δ real devices is $S_\delta = p_0^\delta \prod_{k=1}^{\delta} \frac{1}{\binom{n+m_2-1}{n-k+1}}$. If the adversary accesses all remaining real devices under this protocol, the probability that they can obtain all real measurements through proactive probing will be at least $S_{all} = S_\delta^{\lceil \frac{n}{\delta} \rceil}$.

B. Evaluation

In Section IV, we divide all virtual nodes in two groups. One group is *accessible* by legitimate applications; we add random communication to accessible virtual nodes such that it is challenging for adversaries to identify real devices by passively monitoring communication pattern. The other group is *inaccessible*, and accessing them triggers the probabilistic dropping protocol.

Effectiveness in RO1. In Figure 19, we show probabilities that an adversary successfully guesses whether a device is real based on randomized requests (see Section IV-A). In our experiment, we issue requests to 95% randomly selected real devices and to accessible virtual nodes, whose ratio to real devices is increased from 1% to 10% (in x-axis). Even with a small accessible-virtual-node to real-device ratio, the probability is always lower than 0.001% (10^{-5}), making it challenging for adversaries to distinguish real devices from virtual ones based on the randomized requests.

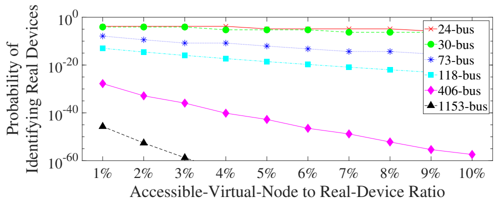


Fig. 19: **Probabilities that an adversary identifies real devices** based on randomized requests.

In Figure 20, we show the accuracy of state estimation when we issue requests to 95% randomly selected real devices and retrieve their physical data. We observed negligible differences, with less than 0.1%. In power grids, state estimation often uses redundant data, e.g., 100% more than necessary data, to ensure estimation accuracy. Even if we use 95% of physical data, the accuracy of state estimation is not affected.

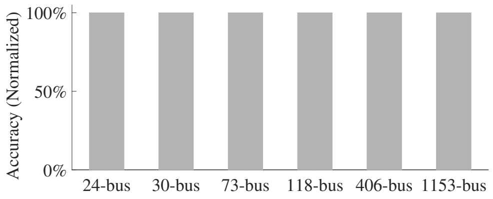


Fig. 20: **The accuracy of state estimation** when we randomly retrieve 95% physical data from real devices.

Effectiveness in RO2. In Figure 21, we show FN rates of the probabilistic dropping protocol, the chances that an adversary successfully obtains measurements of real devices by proactive probing. In this experiment, we set design parameters as $\delta = 4$ and $p_0 = 0.18$. From the figure, we can see that the chance to obtain all measurements through proactive probing is low, less than 10^{-30} for small- or medium-size power systems. Because FN rates are lower than 10^{-200} for large power systems, we did not include the rates in the figure.

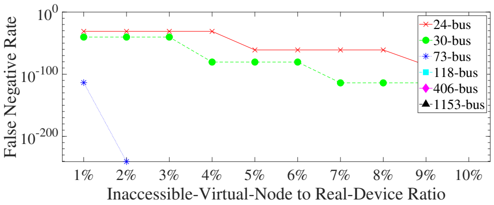


Fig. 21: **False negative rate** of the probabilistic dropping protocol with $\delta = 4$ and $p_0 = 0.18$.

Because only accesses to inaccessible virtual nodes trigger the probabilistic dropping protocol, *there are no false positives (FP) for legitimate applications*, which already know identities of real devices. In rare cases, faulty devices (physical devices used by power grids usually have the probability of misconfiguration between 30×10^{-6} to 600×10^{-6} [55]) or devices unaware of DefRec can accidentally send requests to *inaccessible* virtual nodes. Assuming that there are n real devices and m_2 inaccessible virtual nodes, the probability of accessing one of the *inaccessible* virtual nodes is $m_2/(n+m_2)$. If we present $m_2 = r \times n$ with $0 < r < 1$, then this probability becomes $r/(1+r)$, which is not related to the size of a power grid. Based on the analysis in Figure 12, we can achieve RO2 with $r \leq 10\%$, which makes the probability of accidentally accessing inaccessible virtual nodes 9.1% or less.

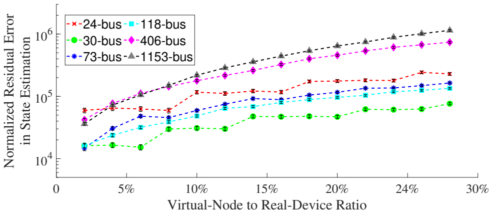


Fig. 22: **Normalized residual errors of state estimation** when adversaries use decoy data to prepare FDIAs (with 99% confidence interval).

Effectiveness in RO3. In Section VII-A3, we demonstrate FP and FN rates of RO3. In Figure 22, we demonstrate the normalized residual errors of state estimation when adversaries use decoy data to prepare FDIAs. As discussed in Section V, if adversaries obtain correct knowledge about power grids, they can design *active* attacks on the FDIAs such that normalized residual errors are smaller than 1. However, if adversaries use decoy data crafted by DefRec, normalized residual errors are amplified to at least 5,000. As we increase the ratio of virtual nodes, adversaries will use more decoy data for attack reconnaissance. Correspondingly, normalized residual errors increase significantly, reaching around 10,000 for all six power grid cases.

Variations in Power Systems. Figure 11 and Figure 12 show that DefRec can delay adversaries for a long latency, e.g., 100 years. In that span of time, operational environments of power grids can experience significant changes. In Figure 23, we show normal variations of real communication networks or transmission networks in power systems from

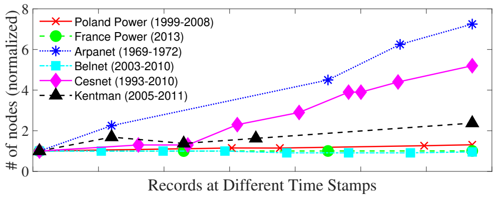


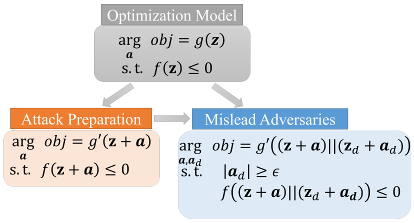
Fig. 23: **Variations in communication networks or power systems:** the x-axis represents a time span specified in the legend during which the data is recorded; the y-axis specifies the number of nodes normalized to the first record in the dataset.

public datasets, including InternetZoo [34], Rocketfuel [66], and MATPOWER [79]. We select six of these networks that contain at least five records at different times.

Because different networks can have various sizes, Figure 23 normalizes the number of nodes in each network with the number appearing in the first record. We can observe significant changes in communication networks, such as Arpanet and Cesnet. In transmission networks used by Polish and French national power grids, we can also see around 10% increase of physical devices in ten years. Variations in a long time interval make it difficult, if not impossible, for adversaries to achieve reconnaissance objectives.

Appendix B
Generalization of Decoy Data Construction

In this section, we show how the decoy data construction procedure designed for FDIAs is applied to other attacks. Most power grid control operations are formulated as an optimization problem. For example, FDIAs aim at minimizing the errors of state estimation while optimal power flow analysis aims at minimizing operational costs [45].



Optimization Model
$\arg_{\boldsymbol{a}} \; obj = g(\mathbf{z})$
s.t. $f(\mathbf{z}) \leq 0$
Attack Preparation
$\arg_{\boldsymbol{a}} \; obj = g'(\mathbf{z} + \boldsymbol{a})$
s.t. $f(\mathbf{z} + \boldsymbol{a}) \leq 0$
Mislead Adversaries
$\arg_{\boldsymbol{a},\boldsymbol{a}_d} \; obj = g'\big((\mathbf{z} + \boldsymbol{a})||(\mathbf{z}_d + \boldsymbol{a}_d)\big)$
s.t. $|\boldsymbol{a}_d| \geq \epsilon$
$f\big((\mathbf{z} + \boldsymbol{a})||(\mathbf{z}_d + \boldsymbol{a}_d)\big) \leq 0$

Fig. 24: **Formulating decoy data construction.** Misleading adversaries into targeting virtual nodes.

In this work, we focus on intelligent adversaries causing physical damage by disrupting control operations, usually formulated in optimization problems. Compared to random disruptions, these attacks, determined based on theoretical analysis of control operations, can introduce severe physical damage without raising alerts [19], [61].

In Figure 24, we present a general format of those optimization problems with the objective specified by g'. In FDIAs, adversaries' objectives are to minimize estimation errors, with compromised measurements leading to wrongly estimated system state (i.e., $\mathbf{z} + \mathbf{a}$ in the figure). In another attack that disrupts optimal power flow analysis, adversaries can maximize the costs of power generation instead of minimizing them, to reduce economical revenues [19], [61].

DefRec mixes decoy data with real one (specified by $\mathbf{z}_d$), such that (i) no solutions exist to achieve attack objectives; and/or (ii) achieving attack objectives requires modifying decoy data with significant changes (indicated by a threshold $|\boldsymbol{a}_d| \geq \epsilon$). In other words, adversaries will prepare attack strategies involving operations of virtual nodes that, when executed, will easily expose the adversaries.

Appendix C
Storage Overhead of Device Profile and Caching

In Table II, we show storage overhead of device profiles. By following meter deployment in [35], we assume that for each power grid, there are two meters measuring the active and reactive power injected at each substation and four meters measuring the active and reactive power flows at the receiving and sending ends of each transmission line. For each meter, we used ten 32-bit numbers to record the range of the measurements and their probability distribution. In Table II, we can see that even for an 1153-bus power grid, we can record a total of 367.8 KB in the machine implementing PFV.

TABLE II: **Storage overhead of device profile:** classified based on power systems and the ratio of virtual nodes to physical devices.

Power Grid	*Base*	*Ratio of Virtual Nodes to Physical Devices*			
		10%	15%	20%	25%
IEEE 24-bus	8.1KB	8.9KB	9.3KB	9.7KB	10.1KB
IEEE 30-bus	9.0KB	9.9KB	10.4KB	10.8KB	11.3KB
RTS96 73-bus	25.1KB	27.6KB	28.9KB	30.1KB	31.4KB
IEEE 118-bus	38.1KB	41.2KB	43.8KB	45.7KB	47.6KB
Poland 406-bus	107.2KB	117.9KB	123.3KB	128.6KB	134.0KB
Poland 1153-bus	301.4KB	331.5KB	346.6KB	361.7KB	367.8KB

TABLE III: **Storage overhead of caching network interactions.**

Supported Function Codes	*Devices*		
	SEL 751A	AB 1400	ION 7550
CONFIRM	✓	✓	✓
READ	✓ (4)	✓ (2)	✓ (2)
WRITE	✓		
SELECT	✓	✓	✓
OPERATE	✓	✓	✓
DIRECT OPERATE	✓	✓	✓
COLD RESTART	✓		✓
WARM RESTART	✓	✓	✓
ENABLE UNSOLICITED	✓		
DISABLE UNSOLICITED	✓		
Total Storage Overhead	$\leq 8KB$	$\leq 5KB$	$\leq 5KB$

In Table III, we present storage overhead of caching network interactions with three physical devices used in our evaluations. We include all network interactions that are supported by those devices and a DNP3 master implemented based on the openDNP3 library [49]. Even though the DNP3 protocol specifies a large amount of data formats, a physical device usually selects a single data format for each function code (see the DNP3 protocol specification for details [29]). Consequently, we cache one request and the corresponding response for each function code. One exception is the "READ" operation, for which we have cached multiple pairs of requests and responses (the amount is included in parentheses). To be compatible with legacy devices, the size of a single DNP3 packet cannot be more than 292 bytes [29]. Consequently, we estimate the total storage overhead for caching network interactions with those three devices, usually occupying less than 8 KB.