The Virtualized Cyber-Physical Testbed for Machine Learning Anomaly Detection: A Wind Powered Grid Case Study

DANIEL L. MARINO¹, CHATHURIKA S. WICKRAMASINGHE¹, VIVEK K. SINGH² JAKE GENTLE², CRAIG RIEGER², AND MILOS MANIC¹ (FIEEE)

¹Virginia Commonwealth University, Richmond, VA. 23220 USA (e-mail: {marinodl, brahmanacsw}@vcu.edu, misko@ieee.org)
²Idaho National Laboratory, Idaho Falls, Idaho, USA (e-mail: craig.rieger@inl.gov)

Corresponding author: Daniel L. Marino (e-mail: marinodl@vcu.edu).

This research was completed by Idaho National Laboratory with funding from the U.S. Department of Energy Wind Energy Technologies Office. Idaho National Laboratory is operated by Battelle Energy Alliance under contract No. DE AC07-05ID14517 as part of the Microgrids, Infrastructure Resilience and Advanced Controls Launchpad (MIRACL) project. This work was supported in part by the Commonwealth Cyber Initiative, an investment in the advancement of cyber R&D, innovation and workforce development. For more information about CCI, visit cyberinitiative.org.

ABSTRACT
Developing tools that help us understand and analyze the effects of cyber attacks on physical assets is necessary in order to detect and prevent harmful consequences of integrating Information and Communication Technologies (ICTs). In this paper, we review existing technologies for developing a fully virtualized cyber-physical testbed for cyber and physical data acquisition and machine learning anomaly detection. We present a testbed that uses network emulation and real industrial communication protocols to emulate the interactions of ICTs inside a wind-powered system. We use the testbed to simulate malicious cyber attacks, their effect on the physical system, and detection mechanisms for such disturbances using anomaly detection. The advantages of the presented virtualized testbed are: 1) integration of real industrial protocols, network analysis tools, and industry-leading data-engineering and machine learning tools; 2) enables a holistic analysis of cyber-physical systems by acquiring and analyzing simultaneously cyber and physical data; 3) cost effective solution for prototyping and testing that can run in a single laptop. The testbed combines unique technologies that enable testing of entire data-driven pipelines, including data acquisition, data management, analysis, and storage, emulating how they would run in a real system. We show how the presented approach can be used to analyze and profile both cyber and physical behavior. The experiments show the capabilities of the presented approach by demonstrating the successful detection of a malicious insertion command through observing anomalous behavior in both cyber and physical data.

INDEX TERMS Cyber-physical systems, Emulation, Machine Learning, Anomaly Detection, Power system security.

I. INTRODUCTION
Modern society has increased reliance on renewable energy sources such as wind energy, which in the United States accounted for 7.3% of generated energy in 2019 [1]. These new energy sources are engineered and constructed with an increased reliance on Information and Communication Technologies (ICTs), which constitute the backbone for communication and control of an increasingly complex network of inter-depandan components [2][3]. ICTs provide the means for increased efficiency, resilience, and cost-effectiveness [2][4]. However, ICTs also open the door to new vulnerabili- ties, making power infrastructure an easy target for various malicious entities. Remote and insecure substations, wind turbines, communication network configurations, and insecure SCADA protocols are some examples that create opportunities for malicious agents. Attackers can access...
critical infrastructure to disrupt the system operation and/or to damage equipment, leading to catastrophic failures and significant financial losses [5]. As a result of these challenges, it is crucial to focus on improving the cyber-physical security and resilience of these systems by implementing methods to detect and prevent possible malicious attacks.

Through the application of anomaly detection, which ingests and evaluates both cyber and physical data sets, a correlation of undesired behaviors and the distinction between malicious and benign can be achieved. The development of these technologies requires careful testing and analysis. Real data is expensive to obtain, especially in cyber security because of privacy and export control concerns. Furthermore, static data does not allow to perform holistic tests of important components such as data acquisition, management, and storage and how they may affect the fidelity of proposed technologies. Cost-effective simulation environments have facilitated recent advances in machine learning and reinforcement learning [6]. This capability is lacking in cyber-security research of cyber-physical systems, with the standard being HIL (Hardware-in-the-Loop) real-time simulations that are prohibitively expensive for many researchers, difficult to maintain, and require extensive training. A cost-effective methodology is necessary in order to evaluate the performance of proposed strategies for machine learning anomaly detection in cyber-physical security. Furthermore, designing a testing methodology that allows integration and tests entire ML pipelines is important in order to enable a holistic analysis of the entire data-driven pipeline as a whole, including how data is acquired, managed, and analyzed in real-time, with real industrial protocols and ML frameworks. This is fundamental in order to allow for testing ML solutions exactly how they will run in real environments, facilitating the transition to production.

Electrical engineers have relied on simulation software for decades in order to develop, improve, and understand the electrical grid under normal and abnormal scenarios. With the increasing introduction of ICTs in the grid infrastructure, it is becoming necessary to introduce models of these technologies in the simulation toolbox available to engineers. Developing a holistic analysis approach that evaluates the impacts of ICTs in Industrial Control Systems (ICSs) is of paramount importance.

This paper presents a fully virtualized testing environment for data-driven analysis of cyber-physical disturbances that simultaneously consider cyber and physical data. The approach provides a testbed environment to run cyber-physical scenarios in a controlled virtualized environment that includes network emulation, industrial communication protocols (DNP3), security tools, and machine learning algorithms. Cyber and physical data is collected from the testbed to perform analysis targeted to anomaly detection. We use a Wind-Farm system for demonstration purposes (Figure 1). The case study includes a distribution network with a wind farm and two loads. Outstations collect data from the grid and the wind farm and transmits it to a data historian using the DNP3 protocol. A malicious device is connected to the system to launch a set of scheduled cyber attacks.

We chose a fully virtualized software-based approach for the simulation/emulation of the cyber-physical system. This provides a cost-effective approach for testing hypotheses and prototyping. Moreover, it is easier to maintain, deploy, share, encapsulate, and cheaper to scale. Because it is fully software-based, standard version control can be used to keep track of changes and ensure repeatability of tests. The testbed provides an efficient approach to test prototypes before committing to more involved tests with HIL, which generally requires expensive equipment that is out of reach for most research worldwide.

Through a series of scenarios, the baselining and recognition capability of cyber-physical disturbances is demonstrated. The scenarios illustrate the benefits of recognizing benign or malicious degradation by the use of both cyber and physical data sets, providing a more comprehensive threat detection. These scenarios are designed to replicate real-world responses that would be expected from typical cyber attacks as compared to potential benign failures, noting that physical protections and cybersecurity are minimal in current implementations. Therefore, the attacks can come from direct physical access to distributed wind turbine compared to a direct attack on a firewall perimeter protection for the entire microgrid network. As current cybersecurity measures used in these applications have neither threat detection nor recognition of foreign assets, we illustrate how the presented environment can be used to evaluate data-driven anomaly detection approaches.

**Objective:** Our objective is to design a fully virtualized testbed that allows us to simultaneously extract cyber and physical data to analyze how cyber and physical components interact and behave under a specific disturbance. More specifically, we are interested in capturing network packet data and sensor data from a fully virtualized cyber-physical testbed. The testbed will enable the development of data-driven technologies for secure and resilient cyber-physical systems.
The rest of the paper is organized as follows: section II presents a review of cyber-physical testbeds, section III presents an overview of the testbed and technologies used, section IV describes the wind-farm case study and the cyber-physical scenarios, section V discusses the experimental results, section VI presents the future opportunities in this context, and section VII concludes the paper.

II. REVIEW OF CYBER-PHYSICAL TESTBEDS

There have been many efforts in developing modern cyber-physical testbeds, which consist of physical operations and communication architectures. For the rest of the paper, we refer exclusively to cyber-physical testbeds in power systems applications. Figure 2 shows the general architecture and components of typical cyber-physical testbeds that will be reviewed in this section. In general, cyber-physical testbeds are composed of three main components: 1) a physical component that simulates/emulates/incorporates a model of the physical system; 2) a cyber component that simulates/emulates/incorporates the network communication and ICTs; 3) an interface that supports the exchange of information and seamless interaction between physical and cyber components. Field devices and Intelligent Electronic Devices (IEDs) sit in between cyber and physical boundaries, using the CP interface to exchange data. Remote hosts that perform supervisory control, data logging, analysis, and remote control use a communication network to interact with all other components. Depending on the specific use-case, cyber and physical components can be simulated, emulated, or directly incorporated in the testbeds. Existing cyber-physical testbeds choose between these technologies in order to find a balance between fidelity and cost-effectiveness.

Emulation is a process that creates an environment that can mimic both the hardware and software configurations and behaviors of real devices such as end hosts, protocol implementations, network links, intermediate nodes, and background traffic [6]. Simulation is a process that creates a mathematical model that can be used to mimic the behavior of real components [7]. A simulation is the result of an abstraction of the real process, while an emulation duplicates the behavior of a device exactly as it behaves in real environments. For practical reasons, physical systems are usually simulated as duplicating their behavior is often not feasible or prohibitively expensive [8]. On the other hand, virtualization technologies provide the tools for emulating cyber components exactly as they behave in real environments [9].

In the following subsections, we review existing cyber-physical testbeds. We separate the review between real-time Hardware-in-the-Loop (HIL) testbeds and fully virtualized testbeds. We choose this distinction because real-time HIL testbeds are the most common type of cyber-physical testbeds in the literature, while a fully virtualized testbed is the goal of this paper. Table 1 shows an overview of the advantages and disadvantages between HIL real-time testbeds and fully virtualized testbeds.

HIL testbeds are characterized by the use of real hardware as the cyber component that interfaces directly with a real-time solver for the physical model (see Fig. 3). HIL testbeds provide the most seamless integration between cyber and physical components. Real-time HIL testbeds rely on the use of real-time simulators as the solver for the physical system model [4][10][16].

Real-time simulators (e.g., RTDS, OpalRT, dSPACE, and Typhoon HIL) provide the required hardware and software to produce calculations within fixed time steps. Real-time simulation is often a requisite for HIL testing as it greatly simplifies the CP interface between real hardware and the physical model [8]. Real-time simulators vendors provide a wide range of interfaces to interact with external hardware, such as digital I/O, analog I/O, field buses, and ethernet.
TABLE 1: Summary of capabilities and drawbacks of real-time HIL vs fully virtualized testbeds (+: limited capability, ++: medium capability, +++: leading capability).

|                         | HIL Real-Time | Virtualized |
|-------------------------|---------------|-------------|
| Cost-effectiveness      | +             | +++         |
| Scalability             | +             | +++         |
| Version control         | +             | +++         |
| Fast prototyping        | ++            | +++         |
| Reconfigurability       | ++            | +++         |
| Determinism (Real-Time) | +++           | +           |
| Physical interfaces     | +++           | +           |

Advantages: Real-time simulations that require deterministic, low latency, and high-frequency feedback loops connected to real hardware through specialized digital and analog IO.

Drawbacks: High cost solution with limited version control and reproducibility, limited flexibility and scalability.

Flexible and scalable cost effective approach. Easier to reconfigure and adapt to test new ideas without depending on any specialized hardware.

Limited real-time simulation capabilities, specially in applications that require deterministic and extremely low latency response using specialized hardware.

FIGURE 3: Typical Real-time HIL testbeds

ports. This wide range of interfaces simplifies the interaction of real industrial hardware with the simulation as they can be directly connected to the simulator. Current commercial real-time simulators provide interfaces primarily for HIL and Controller in the Loop experimentation. However, they are rather limited in their communication interfaces [17].

A. REAL-TIME HIL TESTBEDS

In [4], authors have introduced Cyber-Physical System Resilience (CPSR) testbed architecture. It consists of flexible communication infrastructure, real-time HIL simulation of power system with real control and protection devices, and network function virtualization through SDN switches. This testbed facilitates the implementation, testing, and validation of power system operation and control algorithms. This system has been tested with various cyber-attack scenarios, including denial-of-service, Address Resolution Protocol (ARP) poisoning, insider threats, and false-data injection. Similar efforts, such as networked-microgrid (NMG) testbed [18], facilitate validation of a distributed microgrid power-sharing strategies by finding the impact of cyberattacks and testing control operations such as protective relays, capacitor bank controllers, tap changers, and voltage regulators [4, 10–12, 19]. The work in [4] follows the architecture depicted in Fig. 3 with the addition of Software-Defined-Networking (SDN) technology to support cloud-based intrusion detection and interaction with real control devices to detect and mitigate cyber attacks.

In [19], authors have developed a wide-area measurement system (WAMS) based cyber-physical system using an RTDS with hardware-in-the-loop simulation. The RTDS is used to emulate electrical machines, controllers, transmission system components, and system load. It also provides a HIL simulation environment. They have used MATLAB, Python, and AutoIt scripts to model their system by integrating commercial control and monitoring devices, software, hardware, and standard network communication protocols. The communication infrastructure in the testbed consists of various network protocols, including MODBUS, Telnet, TFTP, DNP3, IEC 61850, and IEEE C37.118. This testbed consists of several substations equipped with PMUs (Phasor Measurement Units). They have achieved substation level control using a HIL configuration using relays. This testbed can simulate five different power system models: a WECC nine bus system, a three generator four bus system, a modified two generator three bus system, IEEE fourteen bus system, and two area power systems.

In [20], researchers have presented the Quasi-Realistic Grid Exercise (QRGridEx), which provides a platform for performing CPS security training and attack-defense exercises. This system was developed in two parts. Part I focuses on modeling the synthetic grid models in RTDS, various cyber attack vectors, and incorporation of anomaly/intrusion detection systems. Part II incorporates risk-based CPS grid investment strategies and incident response management systems. This system has also incorporated various components such as open-source and customized attack libraries (MITRE ATT&CK, NVD/CVE), grid failure scenarios by NESCOR, and standard best practices NIST C2M2, cyber-defense tools, and potential mitigation strategies. They have used the IEEE C37.118 protocol for exchanging data and IEC 61850 GOOSE for communication.

In [21], authors propose an attack-resilient wide-area damping control system (WADC). They have proposed a signal-entropy and physics-based feature extraction approach with an anomaly detection algorithm. They have demonstrated their approach using a hardware-in-the-loop (HIL)
WADC integrated with ADM for a cyber-physical closed-loop test power system. For synchrophasor data communication, they have used the IEEE-C37.118 protocol. They have used DNP3 and IEC-61850 protocols for the SCADA data and control signal communication across the actuators, control center, intelligent electronic devices (IEDs), and remote terminal units (RTUs).

In [22], the authors present a software development framework integrated with a HIL testbed for conducting cyber attack experiments. The hardware architecture of this system consists of simulation machines that provide the embedded computers the ability to sense and control the (simulated) physical systems. The communication network emulator of the system consists of an OpenFlow switch. The software platform of this system is designed using the ROSMOD software tool suite, which provides functions for developing CPS code, attack and measurement code, deploying the experiment, configuring the experiment, and retrieving the results from the experiment. This system implements various attack vectors, including packet snifing, password/authentication, session hijacking, replay, spoofing, packet delay, denial of service, distributed denial of service, and packet dropping attacks. This system also provides attack linking capability, allowing to stage multiple attacks together.

Wind energy is becoming a predominantly renewable energy source. The increased dependence on wind power plants draws the attackers’ attention resulting in cyber security concerns. Therefore, it is necessary to focus on developing testbeds to facilitate implementation, testing, and validation of operations related to wind generation to evaluate the performance under different cyber-physical attack conditions. Existing testbed designs can be leveraged together with the integration of application-specific attributes from wind power generation, such as wind farm threats, geographic scale, utility scale, turbine configurations, protocols, and wind conditions [5, 23, 24]. In [25], a hierarchical network architecture was proposed for large scale wind farms. Ongoing work with Idaho National Laboratory and Sandia National Laboratory has a virtualized implementation of a wind environment for the express purposes of evaluating cyber hardening [26].

Real-time simulators increase the fidelity of the testbed, provide better interfaces for HIL testing, and enable simulations at very low time-steps [27]. However, real-time HIL testbeds come at a considerable price, especially when testing at large scales as they require purchasing all the required hardware. The excessive cost of these simulators means that they are only available to a handful of researches around the world [28]. Real-time HIL testbeds also bring several logistic complexities as people have to share limited hardware, they are harder to maintain, require specialized software and training, are difficult to upgrade, and tests are difficult to replicate as version control is more challenging [29].

Current real-time HIL simulators as depicted in Figure 3 require hardware interfaces (e.g. NICs) in order to interact with external devices. These interfaces are expensive and there is a limit in the number of interfaces that a simulator can host. This poses a scalability issue as there is a limit in the input/output hardware interfaces which limits the number of devices that can interact with the solver. In order to increase scalability without incurring in excessive costs, network emulation is being explored by wrapping the real-time simulator with a solver interface (see Fig. 2) that serves the data from the solver to a network of devices that can be scaled at a lower cost. In [17] a network of Raspberry Pi connected to a real-time simulator (Opal-RT) is used to emulate a distributed control system. A data server was implemented to interface with the real-time simulator. The data server uses UDP to communicate the data from the simulation to the network of Raspberry Pi.

B. FULLY VIRTUALIZED TESTBEDS

In this paper, we focus on the development of a fully virtualized testbed for cyber-physical research. Because of the drawbacks of full HIL simulation, fully virtualized software-based testbeds have been developed in parallel [20–33]. Emulation provides enough fidelity and resemblance to real-world settings, covering an increasingly larger spectrum of applications and test-case scenarios.

Figure 2 presents all the necessary components for a fully virtualized testbed. Hypervisors and containers can be used to emulate networks of cyber components. Simulators such as Simulink, OpenDSS, or PowerWorld simulate the physical system model. The most challenging part is the development of the CP interface between the virtual cyber network and the physical simulation.

Advancements in virtualization technologies have brought a rich toolbox for the virtualization of cyber components. Virtualizations allow to model and execute several isolated cyber components within the same hardware. Type 1 and type 2 hypervisors provide the base virtualization tool to isolate virtual components running in the same hardware. Hypervisors and virtual machines perform hardware virtualization, which allows to create and isolate software as if it were running in separate hardware. Together with network virtualization technologies such as the Linux bridge or the Open vSwitch project, several virtual machines can be created and connected to emulate a fully virtualized computer network.

Hypervisors provide a powerful approach to virtually emulate the cyber components of the cyber-physical testbed. In [34] a virtual testbed is presented, where an OpenPLC [35] instance is launched inside a Linux virtual machine and connected to a Simulink model. Another technology that provides an approach for virtualizing cyber components is OS-level virtualization. OS-level virtualization provides a lightweight alternative to virtual machines. Linux namespaces and containers (e.g., Docker) implement various levels of OS virtualization that provide different levels of isolation of virtual cyber components. These technologies can be used to emulate a network of virtual components communicating using real protocols without requiring changes in code or configuration [9].
Mininet [9] is a software defined network (SDN) emulation tool that provides a low-cost solution for researchers to perform experimentation on SDN functionalities such as prototyping large scalable networks and control forwarding and routing data packets through networks [42]. It supports different programming languages and standard operating systems. It also allows to the creation of scalable networks with thousands of switches in a single computer. Mininet prototypes can use in real networks in real-time, simulating real-time network behaviors [32]. Mininet creates a virtual network by using different network namespaces. Container-net extended Mininet to allow the use of containers in order to provide further isolation of cyber components [43].

In [30], Mininet and PowerWorld are used to study the impact of SDN controller faults on smart grid communication delays. Mininet is used for network emulation, and PowerWorld is used to simulate the physical model. DSSNet [31],[32] is another platform that uses Mininet to emulate the cyber network while using OpenDSS to simulate the physical model.

Perhaps the most challenging component of a fully virtualized testbed is the CP interaction component. HIL testbeds simplify this interaction by running simulations in real-time machines with enough physical peripherals to connect cyber components. For a fully virtualized testbed, there are two main challenges: 1) the transfer of data between several cyber components, 2) the time synchronization of the physical simulation with the cyber emulation.

Transfer of data from the physical simulation is performed by wrapping the simulator with a solver interface that acquires the data from simulation and introduces control signals (see Figure 2). A common approach is to use messaging applications that use TCP or UDP as the communication channel with the solver. [34] uses UDP packets to communicate the Simulink model with a virtualized PLC. [31],[32] use the ZeroMQ messaging library as the communication

| Cyber simulation / emulation | Physical simulation | Cyber-physical interface | Differentiation |
|------------------------------|---------------------|-------------------------|-----------------|
| OPNET | Opal-RT | PC running OPNET connected with an ethernet network card to Opal-RT using UDP protocol | Co-simulation platform using OPNET and OPAL-RT. |
| NetEm, Raspberry PI | Opal-RT | Data server on the Opal-RT with two internal buffers (input/output) that act as a gateway into the physical system | A framework which is built around a fleet of low-cost single board computers (Raspberry Pis), connected to a real-time simulator (OPAL-RT). OPAL-RT based communication interface which allow external controllers to interact with the physical system. |
| OMNeT++ | Universal Environment for Federation (UCEF) | IEEE 1516-2010 High Level Architecture (HLA), a standard for the co-simulation of distributed processes | An approach to integrate a network simulator with the High Level Architecture (HLA). |
| NS-3, OMNeT++/INET | Modelica | DEVS wrappers for the FMI (Functional Mock-up Interface) standard | A middleware dedicated to the co-simulation of CPS based on the Discrete Event System Specification (DEVS) standard. |
| NS-2 | Modelica | Named-pipes communicating application level ns-2 agents and a MIMO (Multiple Inputs Multiple Outputs) modelica model. | Co-simulation platform for CPSs that integrates ns-2 and Modelica. Enables Modelica to respond to internal asynchronous events. |
| QualNet, OMNeT++ | Matlab/Simulink | Custom protocol over TCP/IP | Accurate synchronization approach between Matlab/Simulink with network simulators (QualNet or OMNeT++) for co-simulation of CPSs. |
| OMNeT++ | OpenDSS | Python, COM server, PHP server | Even synchronization uses a Master/Slave approach, where OMNeT++ acts as the Master and OpenDSS works as the Slave. |
| Mininet | OpenDSS | ZeroMQ, named-pipes, COM port | Introduction of Virtual-Time to provide synchronization events between cyber and physical simulations. |
| Mininet | PowerWorld | RPyC | Virtualized testbed that uses Mininet to study the impacts of SDN controllers on AGC (automatic gain control). |
| OpenPLC | Matlab/Simulink | Data transfer through UDP | Fully virtualized testbed that connects a virtualized PLC (OpenPLC) to a Simulink model of a Gas Pipeline. |
| Ours | Mininet, OpenDNP3, Kafka | Matlab/Simulink | ROS2/DDS/RTPS | Fully virtualized testbed for the development of data-driven cyber and physical anomaly detection systems. We use ROS2 to facilitate the backend communication between Simulink and several emulated field devices. We integrate real industrial protocols for physical data collection, real-time network packet analysis, and real data engineering frameworks. |

TABLE 2: Comparison of different cyber-physical testbeds designs, with a focus on designs with virtualized network simulations/emulations.
channel to the physical simulator while named pipes are used to distribute the data to all virtual devices. [30] use RPyC as interface. Testbeds that use OpenDSS use the COM port to interact with the simulation. The use of TCP, UDP, or similar channels between cyber and physical components introduces an unwanted latency between physical and cyber. However, this limitation is usually alleviated by incorporating any real-time logic as part of the physical simulation model, while the logic that depends on network communications can be placed in a virtualized device.

Time synchronization is a problem when the physical simulation runs at a different speed than the virtualized cyber network. Virtual time has been introduced in virtualized networks in order to control the execution and time dilation of the emulated cyber processes. Virtual time have two main advantages: 1) dilation of time when excessive computational loads overwhelm the computational resources, 2) synchronization of physical simulation with emulated cyber devices. In [31][32] a network coordinator is used to interface with the simulator and the virtual time system to control the virtual clock of the simulation. The virtual components can slow down, speed up, or stop the clock to synchronize with the simulator. The virtual time system is particularly useful to allow time-critical processes to be emulated outside the physical simulation. VT-Mininet [44] is an extension of Mininet where virtual time is incorporated to increase simulation fidelity in cases of high workloads.

An alternative to the use of virtualized networks is the use of discrete-event network simulators to model the cyber network in the cyber-physical testbed. Discrete-event network simulators include ns-2, ns-3, OMS++, and OPNET. These simulators allow for prototyping and evaluating cyber-physical scenarios, but they lack the realism of network emulation [9]. A cyber-physical co-simulator is presented in [41] using OpenDSS to simulate the power network, OMNeT++ to simulate the communication network, and a PHP Server and Python scripts to establish the interoperability of these software tools. The IEEE 4-bus test system integrated with a communication network is used to validate the implementation. In contrast to existing software-based testbeds [30][33], the design of our testbed has a particular focus on enabling data analysis and machine learning for both cyber and physical measurements. Previous works have introduced network emulation tools for analysis of cyber-physical systems. For example, authors in [31][32] used mininet and DSS for simulation of power systems. Our work extends existing approaches by embedding data engineering and machine learning tools in the testbed for holistic analysis of cyber and physical disturbances. Our work focuses on acquiring data for data analysis and machine learning applied to detecting cyber and physical disturbances, especially malicious cyber-attacks.

Table 2 shows a comparison of different cyber-physical testbeds designs. To the best of our knowledge, our testbed is the first fully virtualized testbed to be designed specifically for cyber and physical anomaly detection, using real-time network packet analysis, real industrial protocols, and real data engineering frameworks for analysis and collection of cyber and physical data. We adopted special considerations for wind power systems that include: 1) using an industrial protocol common to wind power applications (DNP3) [5]; 2) we implemented disturbance scenarios based on real attacks previously reported on wind farms (attacker modifying the pitch angle of the turbine [5]); 3) leverage the libraries and benchmark models from MATLAB/Simulink specifically developed for wind power systems. The presented testbed allows us to perform holistic design and testing of entire data-driven pipelines that include collecting, managing, and analyzing cyber and physical data. With the exception of field devices that interact directly with the physical system, the testbed allows transferring software from emulated cyber devices directly to real-world applications without requiring any modification in the code.

III. FULLY VIRTUALIZED TESTBED DESIGN

This section presents a general overview of the developed testbed environment with an overview of the technologies used for a fully virtualized Cyber-Physical testbed. Figure 4 presents a general overview of the developed testbed environment. It consists of two main components: a solver that simulates the physical model and a network emulator that emulates the cyber interaction between ICT components using real industrial protocols. The objective of the testbed is to provide an environment for both cyber and physical data analysis.

The developed testbed shown in Figure 4 runs in a single Linux computer. The testbed is composed of two virtual machines (VMs): 1) a Physical VM which runs the physical model simulation, 2) a Cyber VM which runs the cyber network emulation. The two VMs are connected through a bridge, specifically an Open vSwitch (OVS) bridge. We use VirtualBox, a type 2 hypervisor, to run the cyber and physical VMs within a Linux Host PC. We chose a type 2 hypervisor as it allows us to run the testbed in an existing Linux installation, minimizing the setup required to run the testbed.

The testbed is used to simulate the effects of cyber attacks launched by malicious devices, which are incorporated into the simulation. The testbed includes an interface between the network emulation and the physical solver that allows the physical solver to send sensor data and receive control signals. The testbed is divided into two networks: the emulated SCADA network and the solver network. The emulated SCADA network is the network that is being emulated. This network uses real industrial protocols such as DNP3. The solver network is used to communicate the solver (Simulink) with the field devices. This network uses the ROS2 publisher-subscriber framework as the backend infrastructure to communicate data from the solver. The testbed includes tools for measuring and storing cyber and physical data. Network analyzers are installed in the virtualized switches to measure...
the network traffic in the emulated SCADA network. Physical data collected by the sensors in the simulation is managed and stored using data-engineering tools.

The presented approach results from an integration of several disciplines, including network security, systems modeling and simulation, data science, and engineering. In this section, we provide a brief overview of the technologies used to help a reader that may not be familiar with all the disciplines involved. Figure 5 presents an overview of the technologies used for the presented approach. From the diagram, we would like to highlight that besides Simulink and Simscape, all software that we used is open-source and freely available.

The fully virtualized testbed is designed to acquire cyber and physical data during normal and abnormal conditions. Figure 6 illustrates the distribution of three functional control levels across the virtualized testbed: field level, direct control, and supervisory level. In Figure 6 the controllers in the direct control level are split between physical simulation and network emulation. Time-critical operations, such as feedback control, are implemented as part of the physical simulation. Operations that interact with the virtualized network and therefore have more relaxed time constraints are implemented as part of the cyber emulation. This allows us to simplify the interface between cyber emulation and physical simulation as latency and synchronization do not affect time-critical operations. Given the latency and randomness of real-world network communications, the operations that depend on ethernet-based communications is usually not time-critical, which allows us to relax assumptions of time-critical operations in the cyber network operations.

A. CYBER EMULATION

Network emulation duplicates the functionality of real ICT computer networks using the same protocols as in real hardware but executed inside a virtual environment. This technique allows us to test software in an environment that closely follows real-world systems without the complexity and costs of hardware implementations. We chose Mininet [9] for network emulation as it provides a lightweight and cost-effective solution to deploy tests that involve several interconnected devices. Mininet allows us to create a fully virtual environment that uses the same software and protocols used in real-world systems. It allows us to acquire cyber data and evaluate the performance of cyber-physical systems in an encapsulated environment capable of running on a single computer.

Figure 4 shows the architecture of the virtualized cyber network. We distinguish two types of cyber devices: 1) field devices that interact with the physical model (simulator), 2) network devices that interact with the physical model indirectly through the emulated virtual network. In Figure 4 field devices and network devices communicate using the DNP3 industrial protocol. Field devices collect data from the physical model and send control commands to the physical model. Some of the network devices include data historians,
attack PCs, supervisory controls to adjust setpoints, etc. Figure 4 shows, in particular, a data historian and an attack PC, which are the devices implemented in our experiments.

We use Mininet to set up the virtual network topology. Figure 4 shows one switch connecting network devices with field devices. Open vSwitches (OVS) are used as virtual switches in the virtualized network. OVS is an open-source multilayer virtual switch that emulates real-world network operations, including VLAN isolation, traffic filtering, monitoring through Netflow, SPAN ports, and automated control through OpenFlow.

The Distributed Network Protocol version 3 (DNP3) is one of the primary engineering protocols for SCADA systems such as electric power grids [45, 46]. The main purpose of this protocol is to facilitate interactions between different outstations and master stations in SCADA systems via serial-line or TCP/IP protocols through encapsulation. It supports concurrent interaction from multiple sites with a large number of interconnected components. It also provides a secure authentication mechanism to ensure the security of DNP3 messages [45]. For this paper, we use OpenDNP3 as the implementation of the DNP3 protocol. This allows us to perform cyber analysis of the communications with real industrial protocols.

B. PHYSICAL SIMULATION

The physical system is simulated using Simulink and Simscape. Simscape electrical provides several solver types to simulate the physical model. These types include Continuous simulation, Discrete simulation, and Phasor simulation. The continuous simulation uses a variable-step integration algorithm to solve the DAE. This method produces accurate results, but it is prohibitive slow for large systems. The discrete simulation uses fixed-step integration algorithms to solve a discretized version of the system. Using discrete simulation improves the simulation speed, but the step size must be carefully set to guarantee accuracy. Phasor simulation is a simplified model that assumes fixed frequency. It is useful when we are primarily concerned about changes in magnitude and phase of voltage and current in the system components.

In order to simplify the interface between cyber emulation and physical emulation, the physical simulation is required to run at a real-time pace. To achieve real-time pacing, the physical model and simulation are configured to run in discrete phasor simulation. Simulink simulation pacing is used to slow down the simulation, approximating real-time behavior. Phasor simulation allows studying larger systems than discrete dynamic simulation. We are primarily interested in correlating cyber-physical events happening at large time scales (seconds to minutes), which fits the conditions for phasor simulation. Phasor simulations provide a good balance between computational cost and functionality. Further, it enables the study of several scenarios such as frequency stability, low-voltage (LV) ride-through, transient stability, short-term voltage stability, long-term voltage stability, unintentional islanding, power system restoration, among others [47].

C. CYBER-PHYSICAL INTERFACE

Field devices are the devices that interact directly with the physical model simulation. They communicate with the physical simulation through the cyber-physical interface (cp interface, see Fig. 4). The cyber-physical interface is composed of the Host bridge, the cyber interface, and the solver interface. As mentioned before, the ROS2 publisher-subscriber framework is used as the communication backend between the solver and the field devices. We chose ROS2 because it is officially supported by MATLAB/Simulink, simplifying the interface with the solver. ROS2 uses DDS-RTPS as its middleware. RTPS (Real Time Publish-Subscribe protocol) provides publisher-subscriber communications over unreliable transports such as UDP. DDS (Data Distribution Service) is a middleware protocol and API standard for low-latency data connectivity targeted at mission-critical IoT applications. DDS-RTPS serves as the basis for discovery, serialization, and transportation of data between nodes. We specifically used eproxima fast-RTPS implementation of the RTPS protocol. DDS-RTPS is a flexible specification that has been used for reliable, high-level systems integration, as well as real-time applications on embedded devices [48]. Although the presented virtualized testbed could have been implemented directly using DDS-RTPS, we chose to use ROS2 as it provided a high-level API interface that simplified development.

A ROS2 node is used in the Simulink solver to enable real-time communication with the simulation. The solver interface is composed of a series of ROS2 publishers and
subscribers that publish sensor data and subscribe to control inputs. While the solver interface serves as the communication layer with the physical simulation, the cyber interface serves as the communication layer with field devices. The solver interface and cyber interface are connected together with the host PC bridge.

The cyber interface distributes the data from the physical ROS2 node to all field devices. The cyber interface consists of a single OVS switch that connects all field devices with the physical simulation. We use Mininet to set up the OVS switch and connect all field devices to the switch. Each field device contains a ROS2 node that connects to the physical interface through the OVS switch and the host bridge. Each device node subscribes to receive sensor data and publishes control data back to the physical interface. To improve the performance of ROS2, we disabled multicast discovery. We also restricted all ROS2 communication to the solver network, completely isolating solver and SCADA networks. Field devices are the only devices that require code modification in order to interact with the physical simulation. All network devices run 100% unmodified code, which can be used to test software that can be directly deployed in real applications. ROS2 allowed us to integrate several field devices seamlessly in the virtualized testbed, simplifying the integration between physical and cyber components.

D. DATA ANALYSIS AND MACHINE LEARNING

The main purpose of the presented approach is to obtain cyber and physical data from the system in order to analyze and understand the cyber and physical behavior of the system under normal and disturbance scenarios. This section introduces the tools introduced in the testbed to record and analyze cyber and physical data.

Network analysis: Network analysis tools are an essential part of network security research. In this paper, we used these tools to capture and analyze network traffic data. Wireshark is used to analyze captured network traffic data. nmap and fping were used to launch reconnaissance attacks, which simulate how an attacker would act when attempting to gain information about the computer network. Finally, Scapy is a library for network packet analysis and manipulation. We use it primarily to analyze network packet data.

The OVS switches used by Mininet allow mirroring packets using SPAN ports for monitoring and analysis of cyber traffic. This process is illustrated in Figure 7. A cyber sensor is connected to an OVS SPAN port, capturing all the traffic passing through the switch. The cyber sensor is just another emulated network device running in the virtualized cyber network. The cyber sensor uses tcpdump to save packet data as pcap files for later offline analysis. For live analysis, packets are captured and dissected using Scapy. Then, packets are separated in windows of one second in order to extract features for Machine Learning anomaly detection. These features include the number of packets, number of hosts, size of packets, among others. Finally, Machine learning Anomaly Detection algorithms are used to identify anomalies in cyber communication. Anomaly detection algorithms include One Class Support Vector Machines (OCSVMs) [2, 49, 50], Gaussian Mixture Models [51], Stochastic Poisson Processes [3, 2]. An in-depth description of the cyber sensor including feature extraction and anomaly detection can be found in [3, 2].

Data engineering: The developed testbed has the advantage that we can incorporate real unmodified code in the virtualized network devices. The testbed is used for the development and evaluation of data management and storage applications for cyber-physical systems. Several data engineering tools are used to perform the analysis on cyber and physical data. Kafka provides the core functionality to build data pipelines, streaming analytics, and mission-critical applications. We use it primarily to manage the streams of data communicated through DNP3. JSON is a lightweight data-interchange format that we use as the primary format for data streams. Numpy and Pandas provide the core functionalities for data analysis and manipulation. Scikit-learn is a library for general-purpose machine-learning. We used it primarily to train and execute data-driven anomaly detection algorithms. Finally, Scapy is also integrated into the set of data engineering tools as it allows us to pre-process packet data.

Figure 8 shows the design of a data historian which collects and analyzes physical data. The data historian uses a

FIGURE 7: Measurement and analysis of cyber data

FIGURE 8: Storage and analysis of physical data
DNP3 master to collect data from all DNP3 outstations. This data is usually sensor data coming from the physical model simulation. The DNP3 master collects the sensor data and transforms it to JSON format to be published to a Kafka topic. Once the data is in Kafka, any other process in the network can have access to the data for storage and analysis. One process inside the data historian obtains the sensor data from Kafka and stores it as CSV files for later analysis. For live analysis, an Anomaly Detection algorithm requests the physical data from Kafka and reports physical anomalies back to Kafka. The anomalies are also logged by a separate process for offline analysis and development.

**Anomaly detection:** Critical infrastructures such as microgrids rely on information and communication technologies (ICT) to communicate and control their inter-dependent components [2][3]. This dependency has made ICT technologies vulnerable to various cyber-physical attacks such as removal of information from communication channels, interception, replacement, malware releases, and physical attacks [2][52]. In order to prevent such attacks, it is crucial to characterize such malicious behaviors. In recent years machine learning has shown a promising capability to identify such anomalous status in these systems [2][55].

Widely used supervised machine learning algorithm for anomaly detection include Random Forest, Decision Trees, Support Vector Machines, genetic Algorithms and supervised artificial neural network models such as Convolutional Neural Networks (CNNs) [2][52][54][55]. However, the major drawback of these methods is that they require labeled data. The labeling process of these data is an expensive, time-consuming process [56]. Further, in SCADA systems, it is challenging to collect labeled data for all possible abnormal behaviors due to the unpredictable nature of these systems. Therefore, unsupervised machine learning algorithm-based anomaly detection has gained tremendous attention.

Widely used unsupervised machine learning algorithms for anomaly detection include One Class Support Vector Machines (OCSVMs) [2][49][50], Gaussian Mixture Models [51], Stochastic Poisson Processes [2][2], Adversarial Networks [55][57] and other Artificial Neural network based methods [49][55]. These models use the data logs available from historical data to build their inference engines [49].

The testbed is specifically designed to acquire cyber and physical data for the development of machine learning anomaly detection. The testbed emulates a supervisory layer where a data historian collects physical data from field devices. The cyber sensor collects packet data and extracts features to train machine learning anomaly detection algorithms. Anomaly detection is used to detect anomalies in both cyber and physical data. The testbed allows us to capture and correlate the cyber and physical anomalies in time. The anomaly detection algorithms are trained using data collected during normal operation. In this paper, we specifically use Autoencoders, OCSVM, LOF, and Isolation Forest machine learning models to detect anomalies. A series of scheduled attacks are used to obtain attack data and verify that the anomaly detection algorithm is able to detect the attacks.

**IV. WIND-FARM CASE-STUDY**

This section presents the cyber architecture, the physical model, and the list of scenarios used to demonstrate the capabilities of the approach for cyber-physical disturbance analysis. The fully virtualized testbed was evaluated by simulating and measuring the effects of cyber-physical disturbances in a wind farm with ICT integration. Figure 9 illustrates the cyber components of the wind-farm case study. Figure 10 presents the physical model of the wind-farm case study.

A model representative of a technical industry wind farm is used for evaluating cyber-physical scenarios. The technical industry model is spread over a wide area, representing longer feeder lengths. The sizing of this model includes 9 kW of wind turbines, 500 kW of load, and an industrial load of 2 MVA, which includes a 1.68 MW induction motor.

**A. CYBER MODEL:**

The network consists of five main hosts connected using a switch (see Fig. 9). The hosts are 1) a data historian, 2) a DNP3 outstation for the Wind Turbine, 3) a DNP3 outstation for data measured from the grid, 4) a cyber sensor, and 5) a malicious device. We use OpenDNP3 as a communication protocol between outstation and a data historian.

The data historian collects and stores data from the outstations. The outstation serves as a remote device that reads data from the field sensors and performs control actions. The outstation provides sensor measurements of the wind turbine to the historian. These measurements include voltage, current, active and reactive power, the speed of the wind turbine, among other physical measurements (see Fig. 9). The wind turbine DNP3 outstation also allows to perform remote manual control of the pitch angle in the wind-farm.

A malicious device connected to the network executes a series of scheduled cyber attacks. The host is connected to the main switch, and it is programmed to launch reconnaissance attacks and execute malicious control commands through DNP3 to affect the physical system. A cyber-sensor...
is connected to the switch to monitor the communication between devices in the network. The sensor collects the data, analyses the data, and stores the data in pcap files. The sensor extracts packet features and window features following the approach presented in [2]. The extracted features are used to train machine learning models for anomaly detection.

**B. PHYSICAL MODEL:**

We used The Wind Farm DFIG Phasor Model provided by MATLAB Simscape Electrical as the basis for our study. The model consists of a wind farm, a distribution system, a motor load, and a resistive load. The wind farm is composed of six Wind turbines with Doubly-Fed Induction Generators (DFIG) (see Fig 10). The model serves as a distribution level case study from a rural area with relatively long line distances.

The wind turbines use doubly-fed induction generators (DFIG), which are a popular configuration for wind turbines. In this configuration, the stator winding is connected directly to the grid while the rotor is connected to an AC/DC/AC converter. The converter allows to efficiently extract energy from the turbine, reduce mechanical stresses, and absorb reactive power. The model uses phasor type simulation, which allows for transient stability studies for long simulation times [58]. The wind farm includes a feedback control for the pitch angle. When the wind speed increases, the pitch angle also increases to limit the mechanical power.

We modify the original model by introducing a remote manual control mode that allows to controlling the pitch angle in the wind-farm. The manual control signals are provided via DNP3. The DNP3 master is able to send commands to a DNP3 oustation which communicates the commands to the physical solver. The manual control simply replaces the feedback controller signal with the pitch angle provided by the DNP3 master device. The pitch angle signal is passed trough a simple low-pass filter to smooth drastic changes in the manual command.

**C. SCENARIOS:**

We designed three scenarios to demonstrate the capabilities of the presented approach. The objective of the scenarios is to demonstrate how the presented approach can be used to study and evaluate the effects of cyber and physical disturbances. Each one of the scenarios falls into one of three general categories: 1) benign, 2) pure cyber disturbance, 3) malicious cyber-physical disturbance. We use this classification as it helps us evaluate the presented approach and gain insights when we have cyber disturbances that do not affect the physical and when we have cyber causing a direct impact on the physical system.

**Baseline scenario:** Normal operation. This scenario serves as a baseline to characterize the behavior of the system under normal operation. For this scenario, we collect data under expected normal circumstances. The wind turbine operates under normal circumstances, with no faults, and with the plant working continuously. We used a squared waveform to simulate the wind oscillating inside a predefined value. We also consider a variation of ±5% in the loads as part of normal behavior.

**Malicious cyber disturbances:** Recognisance attacks. A malicious device is connected to the switch to launch attacks with the objective of gaining information about the system. A series of attacks are executed to scan the network to identify available hosts. Three attacks are launched one after the other: IP scan, ping sweep, and port scan. The attacks are launched using Nmap and hping3.

**Malicious cyber-physical disturbance:** Pitch angle attack. The malicious device sends a DNP3 command that enables manual mode and sets the pitch angle to zero. This results in an increase of turbine speed above nominal level, leading to increased wear and tear and potential destruction of the turbine. A malicious device can optionally set the Pitch angle to stay just below the maximum operating point to prevent detection and avoid the activation of the turbine protection.

**FIGURE 10: Wind farm DFIG Simulink physical model [58]**
V. EXPERIMENTS
This section discusses the validation of the testbed and the results of machine learning anomaly detection systems tested on the scenarios: baseline, cyber, and cyber-physical scenarios on the testbed.

A. TESTBED VALIDATION
The performance of the testbed was evaluated using the network setup shown in Fig. 11. The setup consists of the Wind DFIG model (Fig. 10) with the ability to replicate the Wind Turbine outstation N times to measure the performance of the testbed with several emulated hosts. The setup simulates the Wind DFIG model with the Wind Turbine outstations reporting data to the data historian every 100ms. The setup also includes two additional emulated devices to measure latency: an echo outstation and a latency test host. The echo outstation forwards messages to an echo function in the Simulink model, which sends the message back to the echo outstation. The latency test host executes two types of tests: Kafka latency test and DNP3 latency test (shown in Fig. 11). The Kafka latency test measures the round-trip time of a message starting from a Kafka publisher in the test host all the way to the echo function in the Simulink model. The DNP3 latency test measures the round-trip time of a message starting directly from a DNP3 master, all the way to the echo function in the Simulink model, without involving Kafka.

Fig. 12 shows the results of DNP3 and Kafka latency relative to the number of devices emulated. We evaluated latency from N=5 up to N=100 emulated devices. The figure shows the mean and an error bar that corresponds to the 0.05 and 0.95 quantiles. The figure shows that the average latency stays below 40ms for both DNP3 and Kafka. As expected, the DNP3 latency is lower than the Kafka latency as the round trip of the DNP3 test does not involve Kafka. On average, the difference between the Kafka latency test and DNP3 is 9.36ms. The figure also shows that the average latency increased with the number of devices. For Kafka, the average latency increased from 35.57ms to 39.71ms when going from 5 to 100 devices. For DNP3, the average latency increased from 27.50ms to 29.25ms. In the case of Kafka, the 0.95 quantile increased considerably from 55.96ms to 74.18ms. Fig. 12 also shows the data throughput in the SCADA network measured in Megabits per second. The throughput increases linearly with the number of devices. This is expected as the devices correspond to a replication of the Wind Turbine outstation, which communicates data every 100ms to the data historian.

Fig. 13 shows additional testing results obtained with the setup from Fig. 11. Fig. 13 shows the bitrate (bandwidth) and packet loss measured with iperf3. The figure shows the results obtained on the SCADA and the solver network for TCP and UDP protocols. For TCP, we observed that the SCADA network is able to handle up to 30 Gbit/sec with five devices, with the bitrate decreasing to 26.77 Gbit/sec with 100 devices. The solver network averaged 1.84 Gbit/sec with five devices, with a steady performance with an increasing number of devices averaging 1.75 Gbit/sec when using 100 devices. The lower bitrate of the solver network, when compared to the SCADA network, is attributed to the connection between the virtual machines. For UDP, we performed a 100Mbps test and an unbounded test. For the UDP 100Mbps test, we observed that both SCADA and solver networks are able to maintain a steady 0.1 Gbit/sec (100Mbps) when increasing the number of devices, with no packet loss in the SCADA network and only a small drop in packets of 0.0018% in the solver network when using 100 devices. For the UDP unbounded test, we observed that the maximum average bitrate in the SCADA network was 3.1 Gbit/sec for five devices, decreasing to 2.69 Gbit/sec when using 100 devices. The solver network achieved an average bitrate of 0.95 Gbit/sec for five devices and 0.79 Gbit/sec for 100 devices. For the SCADA network, we observed a packet loss averaging 0.076%. The Solver network had a slightly higher packet loss, averaging 0.362%. This figure illustrates the capacity of the testbed to handle workloads with multiple emulated devices, with a network bandwidth of around 1Gbit/sec for the solver network and up to 30Gbit/sec for the SCADA network. Compared to the throughput results from Fig. 12 which show a maximum throughput of 2Mbps when using 100 devices, Fig. 13 shows that the virtualized testbed has enough bandwidth to handle the emulation of the presented SCADA network.

Fig. 14 shows the pacing error of the physical simulation. The error is computed as the absolute difference between the system clock and the physical simulation clock. The figure shows values inside the 0.99 quantile. The median of the error is 5.18ms. The pacing error is very high for high-frequency applications that require deterministic real-time requirements. 

![FIGURE 11: Testbed validation](image-url)
time control, e.g., power electronics control. However, the error is acceptable for applications like SCADA systems with sample frequencies in the order of seconds or hundreds of milliseconds, which is the target for which the presented virtualized testbed was designed.

In order to compare the performance of the presented virtualized testbed with respect to real SCADA systems, we replicated the latency tests from Fig. 11 in a real-time HIL testbed. The diagram of the HIL testbed used for testing is presented in Fig. 15. This HIL testbed provides a high-fidelity simulation environment by integrating industry-grade hardware and real-time simulators while emulating grid communication network as close to the real world. We have modeled the Wind Farm DFIG system in ARTEMiS-SSN (State-Space Nodal) solver in eMEGASIM (RT-Lab) environment and simulated using OP5707 OPAL-RT real-time digital simulator at a smaller time step of 50 µs. The modeled DNP3 drivers, inside the simulator, emulate as DNP3 servers and sends data points to the SEL 3555 Real-Time Automation Controller (RTAC) representing the DNP3 client. The RTAC receives the data points and forwards them to the Kafka database.

The HIL testbed uses an Opal-RT to perform the Wind DFIG simulation and the echo function from figure 11. The outstations from the virtual testbed were replaced by DNP3 servers located in the Opal-RT and in the Real-Time Automation Controller (RTAC). Each outstation runs in a different port, with a single Wind Turbine outstation used for the tests. We considered two setups for the real-time HIL test; an Opal-RT setup that only involves the Opal-RT and an RTAC setup that involves both RTAC and Opal-RT. Similar to the tests performed in the virtualized testbed (Fig. 12), we performed round-trip Kafka latency and DNP3 latency tests. Fig. 16 shows a comparison of the latency results obtained with the virtualized testbed, the Opal-RT setup, and the RTAC setup. As expected, the tests directly communicating with the Opal-RT have lower latency than the tests that involve the RTAC. With the Opal-RT, DNP3 latency averaged 493.43 ms. When using an RTAC, the DNP3 latency test averages 533.46 ms, an increment of 40 ms with respect to the Opal-RT setup. Kafka round-trip latency averages 513.68 ms for the Opal-RT setup, an increment of 20.25 ms with respect to the DNP3 test. Interestingly, Kafka introduces lower latency than the RTAC.

For the RTAC setup, Kafka’s round-trip latency averages 592.23 ms, an increment of 58.77 ms with respect to the DNP3 test. With the Opal-RT, DNP3 latency averaged 493.43 ms. When using an RTAC, the DNP3 latency test averages 533.46 ms, an increment of 40 ms with respect to the Opal-RT setup. Kafka’s round-trip latency averages 513.68 ms for the Opal-RT setup, an increment of 20.25 ms with respect to the DNP3 test. Interestingly, Kafka introduces lower latency than the RTAC. For the RTAC setup, Kafka’s round-trip latency averages 592.23 ms, an increment of 58.77 ms with respect to the DNP3 test.
B. CYBER AND PHYSICAL ANOMALY DETECTION

In this section, we demonstrate the applicability of the presented virtualized testbed for the development of cyber and physical anomaly detection systems. This section discusses the results obtained after running the baseline, cyber, and cyber-physical scenarios on the testbed alongside the results of the cyber and physical anomaly detection systems.

Data collection and training of ADS models: The presented virtualized testbed was used to simulate the scenarios presented in section IV-C. Cyber and physical data is collected during the simulation of these scenarios which consist of a series of normal and disturbance events. The data collected from these scenarios is used to train and test a series of ML anomaly detection models. Physical data was collected directly from the emulated data-historian while cyber data was collected from the emulated cyber-sensor. The virtualized testbed allowed us to emulate the data collection and real-time analysis as it would be executed in a real system. All data is collected using the virtualized testbed with the SCADA network configuration shown in Fig. 11. Data was collected from four independent runs of each of the three scenarios, with loads randomly set at \( \pm 5\% \) of the nominal value. We used windows of 1 second to aggregate physical data. The cyber sensor also used windows of 1 second to group network packet data and extract the cyber features presented in [2]. The dataset collected was composed of a total of 4979 physical records and 4416 cyber records.

The cyber and physical ADS models are trained using the data collected during the baseline scenario simulated with the virtualized testbed. ADS models are trained using only data from the baseline scenario. We select one of the independent runs of the baseline scenario for testing while the remaining runs are used for training. The data from the cyber and cyber-physical disturbance scenarios are used as part of the testing data. The objective of the ADS models is to detect any anomalous data that deviates from the baseline. Thus, the ADS system will output 0 when the system operates similar to the baseline behavior on which ADS is trained on. ADS outputs 1 when the system operates in behavior that deviates from the baseline, indicating abnormal behaviors.

Figure 17 shows the cyber and physical measurements obtained during the normal operation of the system. Figure 17a shows the physical values of the sensor measurements read by the DNP3 outstation and sent through DNP3 to be collected in the Kafka topic at the data historian. The sensor...
measurements include voltage and power measured at B575 (see Fig. [10]), the DC voltage at the turbine AC/DC/AC converter (Vdc), and the speed of the wind turbine. Figure [17a] shows the physical values have a periodic behavior, which is the result of the periodic square wave used to simulate wind speed.

Figure [17b] shows the cyber measurements captured during the normal operation of the system. The figure shows the number of packets communicated between devices in the form of an adjacency matrix. The figure also shows the packet rate over time. We observe that the communication is characterized by a low packet rate with very low variation across the entire experiment.

![Figure 17: Sample of cyber and physical data collected using the virtualized testbed. This data is obtained from the emulated data-historian and cyber-sensor. The data is used to train and test anomaly detection systems.](image)

**Baseline scenario:** Fig. [18] shows an example of the cyber and physical measurements obtained during the cyber disturbance scenario alongside the output from the ADS. Similar to the previous experiment, we picked the turbine speed and packet rate as representative examples of the data collected during the experiments. The turbine speed is a subset of the physical data collected, while the packet rate is a subset of the cyber data collected. Fig. [18] shows the behavior of these two variables during the normal operation of the system. The figure shows how the speed of the turbine (Speed(pu)) is limited up to around 1.2 pu. This is the result of the pitch control, which increases the pitch angle to slow down the turbine.

Baseline data were used to train cyber and physical ADS models. The output of the trained ADS models is displayed in figure [18]. Because this data represents the system’s normal operation, the output of both cyber and physical ADS is zero, indicating that there is no anomaly. Fig. [18] shows results using Autoencoders (AE) for both cyber and physical ADS.

**Cyber intrusion:** Fig. [19] shows an example of the cyber and physical measurements obtained during the cyber disturbance scenario along with the output from the ADS. Similar to the previous experiment, we picked the turbine speed and packet rate as representative examples of the data collected during this experiment. Figure [19] shows the behavior of these two variables during the cyber disturbance. Because this data represents a cyber disturbance, the output of the physical ADS is zero (orange line of top sub plot), indicating that there is no effect from these executed cyber attacks on the physical behavior of the system. Thus, as similar to the normal physical behavior presented in Figure [18] (blue line of top sub plot), the turbine (Speed(pu)) is limited up to around 1.2 pu. However, when looking at the behavior of packet rate (blue line, bottom sub plot), we can see high packet rates during all the attack scenarios (IP scan, port scan, ping sweep) compared to the expected normal behavior of packet rate of the system presented in Figure [18]. Further, cyber ADS outputs the value 1 indicating these cyberattack scenarios (orange line of the bottom sub plot).

Fig. [20] shows the connections detected during the intrusion scenario. The figure shows the packet rate between devices in the format of an adjacency matrix. When compared with the data obtained during normal operation (Fig. [17b]), the intrusion scenario shows the communication with an additional device that corresponds to the attacker. These connections were not present during normal operation and
are the first sign of anomalous behavior.

**Cyber-physical attack:** Fig. 21 shows an example of the cyber and physical measurements obtained during the cyber-physical disturbance scenario alongside the output from the ADS. As same with previous scenarios, we selected the turbine speed and packet rate as representative examples of the data collected during the experiments. Thus, Figure 21 shows the behavior of these two variables during the cyber-physical disturbance. In this experiment, we executed IP scan and DNP3 command insertion attacks. When looking at the physical behavior (turbine speed, blue line of top sub plot), we can see that the IP scan has not affected the normal physical behavior, indicating a turbine speed up to 1.2 pu. However, during DNP3 command insertion, we can see a clear increase of turbine speed (1.3 pu) compared to the normal turbine speed, which we obtained during the system’s normal operation (1.2 pu). The physical ADS indicates this behavior by showing the value of one during DNP3 command insertions (orange line of top sub plot). When looking at the cyber behavior (packet rate, blue line of the bottom sub plot), we can see high packet rates during both IP scan and DNP3 command insertion attacks, compared to the packet rate of the normal operation behavior of the system presented in Figure 21. Further, cyber ADS shows the value 1 indicating these cyber-physical disturbance scenarios (orange line of bottom sub plot).

**Comparison of ADS algorithms:** Table 3 shows False Positive Rates (FPR) and Disturbance Anomaly Rates (DAR) for selected anomaly detection algorithms: Isolation Forest (IFo), Local Outlier Factor (LOF), One-Class SVM (OCSVM), and Autoencoders (AE). The metrics measure the performance of the models to detect anomalies in the cyber and cyber-physical disturbance scenarios when trained only with the baseline scenario data. DAR indicates the rate of reported anomalies during disturbance scenarios. Higher DAR indicates better model performance. FPR indicates the percentage of normal records identified incorrectly as abnormalities. Lower FPR indicates better model performance. We report performance on testing data, with results evaluated using k-fold cross-validation where one of the baseline runs is picked for testing while the rest is used for training. A total of four rounds of k-fold cross-validation were used to measure average performance on testing data. The table shows the results of individual cyber and physical anomaly detection before the metric calculation. The results show that AE provided the lowest FPR for both cyber and physical ADSs. Although OCSVM provided slightly higher DAR than AE, lower FPR are preferred over slightly higher DAR. The reason is that the baseline scenarios are composed entirely by normal data, hence FPR should be ideally zero. On the other hand, disturbance scenarios have some overlapping normal behavior during the execution, hence the DAR does not necessarily needs to be 1. An example of this observation...
### VI. FUTURE OPPORTUNITIES

The Internet-of-Things (IoT) revolution has energized the development of virtualized environments. Among the technologies in development that could greatly benefit the presented virtualized testbed is incorporating virtualized real-time (deterministic) devices. The Preempt-RT is a patch of the Linux kernel that enables preemption of low priority tasks in order to ensure the reaction to external events (e.g. interrupts) within a predefined time frame. Currently, Preempt-RT is available for every long-term stable version of the Linux kernel since version 2.6.11. Preempt-RT is being supported by the Linux Foundation and it is planned to be merged with the main branch of the Linux kernel \[59\]. This patch could allow the incorporation of real-time devices in the testbed running on Linux. Project ACRN is developing a hypervisor for IoT embedded applications development. The project is built with real-time and safety-criticality in mind \[60\]. ACRN allows the deployment of Real-Time Virtual machines, providing a framework for the execution of virtualized real-time devices.

### VII. CONCLUSION

In this paper we presented a review of technologies for the development of fully virtualized cyber-physical testbeds. Based on these technologies, we presented a fully virtualized testbed design that incorporates solutions from power engineering, computer networks, virtualization, data engineering, and machine learning.

The presented approach allows us to run a set of predefined scenarios and record cyber and physical data from the virtualized testbed. We use the acquired data to study and characterize the system’s behavior under three scenarios: normal operations, malicious cyber attacks, and malicious cyber-physical attacks. We demonstrate that the proposed testbed allows us to study and characterize the system by looking at physical data, cyber data, and their interactions. The study that we present follows a malicious device launching a series of scheduled attacks. These include reconnaissance attacks and malicious command insertions.

The results show that the presented approach provides integrated analysis and profiling tools for cyber and physical behavior. Moreover, it shows how cyber measurements can be used to identify anomalous malicious behavior that targets the communication network. Further, anomaly detection in physical data successfully identified anomalous behavior after the execution of a malicious remote control operation. The anomaly detection in physical data was complemented by the detection of anomalous behavior in the cyber measurements, providing clear evidence that the presented approach can be used to profile cyber and physical disturbance behavior.

### REFERENCES

1. U. E. I. Administration, *Wind explained: Electricity generation from wind*. [Online]. Available: https://www.eia.gov/energyexplained/wind/electricity-generation-from-wind.php.
2. D. L. Marino, C. S. Wickramasinghe, K. Amarasinghe, H. Challa, P. Richardson, A. A. Jillepalli, B. K. Johnson, C. Rieger, and M. Manic, “Cyber and physical anomaly detection in smart-grids,” in *2019 Resilience Week* (RWS), vol. 1, 2019, pp. 187–193.
3. D. L. Marino, C. S. Wickramasinghe, C. Rieger, and M. Manic, “Data-driven stochastic anomaly detection on smart-grid communications using mixture poisson distributions,” in *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1, 2019, pp. 5855–5861.
4. M. M. S. Khan, A. Palomino, J. Brugman, J. Giraldo, S. K. Kasera, and M. Parvania, “The cyberphysical power system resilience testbed: Architecture and applications,” *Computer*, vol. 53, no. 5, pp. 44–54, 2020. DOI: 10.1109/MC.2020.2973107.
5. J. Staggs, D. Ferlemann, and S. Shenoi, “Wind farm security: Attack surface, targets, scenarios and mitigation,” *International Journal of Critical Infrastructure Protection*, vol. 17, pp. 3–14, 2017.
A. Hahn, A. Ashok, S. Sridhar, and M. Govindarasu, D. Bullock, B. Johnson, R. B. Wells, M. Kyte, and L. Ren, Y. Qin, B. Wang, P. Zhang, P. B. Luh, G. Cao, W. Gu, C. Gu, W. Sheng, J. Pan, R. Li, and L. A. Pratt, M. Baggu, F. Ding, S. Veda, I. Mendoza, and G. Cao, W. Gu, C. Gu, W. Sheng, J. Pan, R. Li, and L. Avalos, A. Zamora, O. Escamilla, and M. R. A. Paternina, “Real-time hardware-in-the-loop implementation for power systems protection,” in 2018 IEEE PES Transmission Distribution Conference and Exhibition - Latin America (T-D-LA), 2018, pp. 1–5.

B. Lantz, B. Heller, and N. McKeown, “A network in a laptop: Rapid prototyping for software-defined networks,” in Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ser. Hotnets-IX, Monterey, California: Association for Computing Machinery, 2010, ISBN: 9781450304092. DOI: [10.1145/1868447.1868466] [Online]. Available: https://doi.org/10.1145/1868447.1868466

A. Avalos, A. Zamora, O. Escamilla, and M. R. A. Paternina, “Real-time hardware-in-the-loop implementation for power systems protection,” in 2018 IEEE PES Transmission Distribution Conference and Exhibition - Latin America (T-D-LA), 2018, pp. 1–5.

G. Cao, W. Gu, C. Gu, W. Sheng, J. Pan, R. Li, and L. Sun, “Real-time cyber-physical system co-simulation testbed for microgrids control,” IET Cyber-Physical Systems: Theory Applications, vol. 4, no. 1, pp. 38–45, 2019.

A. Pratt, M. Baggu, F. Ding, S. Veda, I. Mendoza, and E. Lightner, “A test bed to evaluate advanced distribution management systems for modern power systems,” in IEEE EUROCON 2019 -8th International Conference on Smart Technologies, 2019, pp. 1–6.

G. Cao, W. Gu, C. Gu, W. Sheng, J. Pan, R. Li, and L. Sun, “Real-time cyber-physical system co-simulation testbed for microgrids control,” IET Cyber-Physical Systems: Theory & Applications, vol. 4, no. 1, pp. 38–45, 2019.

L. Ren, Y. Qin, B. Wang, P. Zhang, P. B. Luh, and R. Jin, “Enabling resilient microgrid through programmable network,” IEEE Transactions on Smart Grid, vol. 8, no. 6, pp. 2826–2836, 2016.

A. Hahn, A. Ashok, S. Sridhar, and M. Govindarasu, “Cyber-physical security testbeds: Architecture, application, and evaluation for smart grid,” IEEE Transactions on Smart Grid, vol. 4, no. 2, pp. 847–855, 2013.

D. Bullock, B. Johnson, R. B. Wells, M. Kyte, and Z. Li, “Hardware-in-the-loop simulation,” Transportation Research Part C: Emerging Technologies, vol. 12, no. 1, pp. 73–89, 2004, ISSN: 0968-090X. DOI: [10.1016/j.trc.2002.02.002] [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X02000792

C. Gavriluta, C. Boudinet, F. Kupzog, A. Gomez-Exposito, and R. Caire, “Cyber-physical framework for emulating distributed control systems in smart grids,” International Journal of Electrical Power & Energy Systems, vol. 114, p. 105375, 2020, ISSN: 0142-0615. DOI: [https://doi.org/10.1016/j.ijepes.2019.06.033] [Online]. Available: https://www.sciencedirect.com/science/article/pii/S014206151931874X

L. Ren, Y. Qin, Y. Li, P. Zhang, B. Wang, P. B. Luh, S. Han, T. Orekan, and T. Gong, “Enabling resilient distributed power sharing in networked microgrids through software defined networking,” Applied Energy, vol. 210, pp. 1251–1265, 2018, ISSN: 0306-2619. DOI: [https://doi.org/10.1016/j.apenergy.2017.06.006] [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306261917307560

U. Adhikari, T. Morris, and S. Pan, “Wams cyber-physical test bed for power system, cybersecurity study, and data mining,” IEEE Transactions on Smart Grid, vol. 8, no. 6, pp. 2744–2753, 2017. DOI: [10.1109/TSG.2016.2537210]

G. Ravikumar, B. Hyder, and M. Govindarasu, “Next-generation cps testbed-based grid exercise - synthetic grid, attack, and defense modeling,” in 2020 Resilience Week (RWS), 2020, pp. 92–98. DOI: [10.1109/RWS50334.2020.9241284]

R. Gelli and G Manimaran, “Anomaly detection and mitigation for wide-area damping control using machine learning,” in 2020 IEEE Power Energy Society General Meeting (PESGM), 2020, pp. 1–1. DOI: [10.1109/PESGM41954.2020.9281615]

B. Potteiger, W. Emfinger, H. Neema, X. Koutoskos, C. Tang, and K. Stouffer, “Evaluating the effects of cyber-attacks on cyber physical systems using a hardware-in-the-loop simulation testbed,” in 2017 Resilience Week (RWS), 2017, pp. 177–183. DOI: [10.1109/RWEEK.2017.8088669]

H. Wu, J. Liu, J. Liu, M. Cui, X. Liu, and H. Gao, “Power grid reliability evaluation considering wind farm cyber security and ramping events,” Applied Sciences, vol. 9, p. 3003, Jul. 2019. DOI: [10.3390/app9153003]

“Roadmap for wind cybersecurity,” Jul. 2020. DOI: [10.2172/1647705]

M. A. Ahmed, “Remote monitoring with hierarchical network architectures for large-scale wind power farms,” Journal of Electrical Engineering and Technology, vol. 10, pp. 1319–1327, May 2015. DOI: [10.5370/JEET.2015.10.3.1319]
[26] J. Gentle and J. Johnson, Cybersecurity for wind energy, Jan. 2020. DOI: 10.13140/RG.2.2.20634.39360 [Online]. Available: https://www.researchgate.net/publication/338924438_Cybersecurity_for_Wind_Energy

[27] R. Krenn and B. Schäfer, “Limitations of hardware-in-the-loop simulations of space robotics dynamics using industrial robots,” 1999.

[28] W. Grega, “Hardware-in-the-loop simulation and its application in control education,” in FIE’99 Frontiers in Education. 29th Annual Frontiers in Education Conference. Designing the Future of Science and Engineering Education. Conference Proceedings (IEEE Cat. No.99CH37011), vol. 2, 1999, 12B6/7–12B612 vol.2.

[29] J. A. Ledin, “Hardware-in-the-loop simulation,” Embedded Systems Programming, vol. 12, pp. 42–62, 1999.

[30] U. Ghosh, X. Dong, R. Tan, Z. Kalbarczyk, D. K. Yau, and R. K. Iyer, “A simulation study on smart grid resilience under software-defined networking controller failures,” in Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security, 2016, pp. 52–58.

[31] C. Hannon, J. Yan, and D. Jin, “Dssnet: A smart grid modeling platform combining electrical power distribution system simulation and software defined networking emulation,” in Proceedings of the 2016 ACM SIGSIM conference on principles of advanced discrete simulation, 2016, pp. 131–142.

[32] C. Hannon, J. Yan, D. Jin, C. Chen, and J. Wang, “Combining simulation and emulation systems for smart grid planning and evaluation,” ACM Transactions on Modeling and Computer Simulation (TOMACS), vol. 28, no. 4, pp. 1–23, 2018.

[33] P. S. Sarker, V Venkataraman, D. S. Cardenas, A Srivastava, A Hahn, and B Miller, “Cyber-physical security and resiliency analysis testbed for critical microgrids with ieee 2030.5,” in 2020 8th Workshop on Modeling and Simulation of Cyber-Physical Energy Systems, IEEE, 2020, pp. 1–6.

[34] T. Alves, R. Das, and T. Morris, “Virtualization of industrial control system testbeds for cybersecurity,” in Proceedings of the 2nd Annual Industrial Control System Security Workshop, ser. ICSS ’16, Los Angeles, CA, USA: Association for Computing Machinery, 2016, 10–14, isbn: 9781450347884. DOI: 10.1145/3018981.3018988 [Online]. Available: https://doi.org/10.1145/3018981.3018988

[35] T. R. Alves, M. Buratto, F. M. de Souza, and T. V. Rodrigues, “Openplc: An open source alternative to automation,” in IEEE Global Humanitarian Technology Conference (GHTC 2014), 2014, pp. 585–589. DOI: 10.1109/GHTC.2014.6970342

[36] D. Bian, M. Kuzlu, M. Pipattanasomporn, S. Rahman, and Y. Wu, “Real-time co-simulation platform using opal-rt and opnet for analyzing smart grid performance,” in 2015 IEEE Power Energy Society General Meeting, 2015, pp. 1–5. DOI: 10.1109/PESGM.2015.7286238

[37] T. Roth, C. Nguyen, M. Burns, and H. Neema, “Integrating a network simulator with the high level architecture for the co-simulation of cyber-physical systems,” 2020. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=929390

[38] B. Camus, T. Paris, J. Vaubourg, Y. Presse, C. Bourjot, L. Ciarletta, and V. Chevrier, “Co-simulation of cyber-physical systems using a devs wrapping strategy in the mecsyco middleware,” SIMULATION, vol. 94, no. 12, pp. 1099–1127, 2018. DOI: 10.1177/0037549717749014. eprint: https://doi.org/10.1177/0037549717749014. [Online]. Available: https://doi.org/10.1177/0037549717749014

[39] A. T. Al-Hammouri, “A comprehensive co-simulation platform for cyber-physical systems,” Computer Communications, vol. 36, no. 1, pp. 8–19, 2012, ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2012.01.003 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366412000047.

[40] A. Suzuki, K. Masutomi, I. Ono, H. Ishii, and T. Onoda, “Cps-sim: Co-simulation for cyber-physical systems with accurate time synchronization,” IFAC-PapersOnLine, vol. 51, no. 23, pp. 70–75, 2018, 7th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2018, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2018.12.013 [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318353456.

[41] G. O. Troiano, H. S. Ferreira, F. C. L. Trindade, and L. F. Ochoa, “Co-simulator of power and communications systems with accurate time synchronization,” in IFAC-PapersOnLine, vol. 94, no. 12, pp. 1099–1127, 2018. DOI: 10.11407/S0140366412000047.

[42] R. L. S. de Oliveira, C. M. Schweitzer, A. A. Shinoda, and Ligia Rodrigues Prete, “Using mininet for emulation and prototyping software-defined networks,” in 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), 2014, pp. 1–6.

[43] M. Peuster, H. Karl, and S. van Rossem, “Medicine: Rapid prototyping of production-ready network services in multi-pop environments,” in 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2016, pp. 148–153. DOI: 10.1109/NFV-SDN.2016.7919490

[44] J. Yan and D. Jin, “Vt-mininet: Virtual-time-enabled mininet for scalable and accurate software-defined network emulation,” in Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research, 2015, pp. 1–7.
[45] R. Amoah, S. Camtepe, and E. Foo, “Securing dnp3 broadcast communications in scada systems,” IEEE Transactions on Industrial Informatics, vol. 12, no. 4, pp. 1474–1485, 2016.

[46] S. East, J. Butts, M. Papa, and S. Shenoi, “A taxonomy of attacks on the dnp3 protocol,” in Critical Infrastructure Protection III, C. Palmer and S. Shenoi, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 67–81.

[47] D. Fetzer, Development of a MATLAB/Simulink Framework for Phasor-Based Power System Simulation and Component Modeling Based on State Machines. kassel university press GmbH, 2018, vol. 7.

[48] W. Woodall, Ros on dds, "https://design.ros2.org/articles/ros_on_dds.html". Accessed: 2021-09-30.

[49] J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, “Anomaly detection for a water treatment system using unsupervised machine learning,” in 2017 IEEE International Conference on Data Mining Workshops (ICDMW), 2017, pp. 1058–1065.

[50] “One-class support vector machines—an application in machine fault detection and classification,” Computers and Industrial Engineering, vol. 48, no. 2, pp. 395 –408, 2005.

[51] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, “Deep autoencoding gaussian mixture model for unsupervised anomaly detection,” in International Conference on Learning Representations, 2018. [Online]. Available: https://openreview.net/forum?id=BJJHLbb0-

[52] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. Hashem, “Attack and anomaly detection in iot sensors in iot sites using machine learning approaches,” Internet of Things, vol. 7, p. 100059, 2019, ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2019.100059. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2542660519300241

[53] C. S. Wickramasinghe, K. Amarasinghe, D. L. Marino, C. Rieger, and M. Manic, “Explainable unsupervised machine learning for cyber-physical systems,” IEEE Access, pp. 1–1, 2021. DOI: 10.1109/ACCESS.2021.3112397

[54] G. Stein, B. Chen, A. S. Wu, and K. A. Hua, “Decision tree classifier for network intrusion detection with ga-based feature selection,” in Proceedings of the 43rd Annual Southeast Regional Conference - Volume 2, ser. ACM-SE 43, Kennesaw, Georgia: Association for Computing Machinery, 2005, 136–141, ISBN: 1595930590. DOI: 10.1145/1167253.1167288. [Online]. Available: https://doi.org/10.1145/1167253.1167288

[55] “Survey on anomaly detection using data mining techniques,” Procedia Computer Science, vol. 60, pp. 708–713, 2015, Knowledge-Based and Intelligent Information & Engineering Systems 19th Annual Con-
VIVEK KUMAR SINGH (Member, IEEE) received the B.E. degree in electrical engineering from the National Institute of Technology Durgapur, India, in 2014, and the M.Eng. and Ph.D. degrees in electrical engineering with a co-major in computer engineering from the Iowa State University (ISU) in 2020. He is currently a Postdoctoral Research Associate with the Department of Power and Energy Systems, Idaho National Laboratory, Idaho Falls, USA. His research interests include cyber-physical security for power system, synchrophasor-based wide-area monitoring, protection, and control (WAMPAC) applications, hybrid energy storage systems, and cyber-physical federation testbeds.

JAKE GENTLE is a senior power systems engineer for Idaho National Laboratory. As a project manager and staff engineer, Gentle develops technology and provides technical oversight by coordinating state-of-the-art and innovative solutions for the U.S. Department of Energy’s Office of Energy Efficiency and Renewable Energy (EERE) and Office of Electricity (OE), U.S. Department of Homeland Security, U.S. Department of Defense, and various electric power industry partners. Gentle previously held the INL Laboratory Relationship Manager position to the Department of Energy’s Wind Energy Technologies Office (WETO).

MILOS MANIC (SM’06-M’04-SM’96) is a Professor with the Computer Science Department and Director of VCU Cybersecurity Center at Virginia Commonwealth University. He completed over 40 research grants in data mining and machine learning applied to cyber security, critical infrastructure protection, energy security, and resilient intelligent control. Dr. Manic has given over 40 invited talks around the world, authored over 200 refereed articles in international journals, books, and conferences, holds several U.S. patents and has won 2018 R&D 100 Award for Autonomic Intelligent Cyber Sensor (AICS), one of top 100 science and technology worldwide innovations in 2018. He is an inductee of US National Academy of Inventors (class of 2019) and a Fellow of Commonwealth Cyber Initiative (specialty in AI & Cybersecurity). He is an IEEE Fellow, recipient of IEEE IES 2019 Anthony J. Hornfeck Service Award, 2012 J. David Irwin Early Career Award, 2017 IEM Best Paper Award and serves as an associate editor of Transactions on Industrial Informatics, Open Journal of Industrial Electronics Society, and is IES Officer and Senior AdCom member. He served as associate editor of Trans. on Industrial Electronics, was a founding chair of IEEE IES Technical Committee on.

CRAIG RIEGER (SM’08) received the B.S. and M.S. degrees in chemical engineering from Montana State University, Bozeman, MT, USA, in 1983 and 1985, respectively, and the Ph.D. degree in engineering and applied science from Idaho State University, Pocatello, ID, USA, in 2008. He is the Chief Control Systems Research Engineer and a Directorate Fellow with Idaho National Laboratory (INL), Idaho Falls, ID, pioneering interdisciplinary research in the area of next-generation resilient control systems. In addition, he has organized and chaired 11 Institute of Electrical and Electronics Engineers technically cosponsored symposia and one National Science Foundation Workshop in this new research area, and authored more than 50 peer-reviewed publications. He has 20 years of software and hardware design experience for process control system upgrades and new installations. He has also been a Supervisor and Technical Lead for control systems engineering groups having design, configuration management, and security responsibilities for several INL nuclear facilities and various control system architectures.